(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
28 February 2002 (28.02.2002)

PCT

(10) International Publication Number
WO 02/17146 A1

(51) International Patent Classification[7]: G06F 17/30

(21) International Application Number: PCT/US01/00468

(22) International Filing Date: 8 January 2001 (08.01.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:

60/174,831	7 January 2000 (07.01.2000)	US
60/174,832	7 January 2000 (07.01.2000)	US
60/174,833	7 January 2000 (07.01.2000)	US
60/174,835	7 January 2000 (07.01.2000)	US
60/174,842	7 January 2000 (07.01.2000)	US
60/187,030	6 March 2000 (06.03.2000)	US

(71) Applicant *(for all designated States except US)*: **ETANTRUM. COM, INC.** [US/US]; 44901 Falcon Place, Suite 114, Dulles, VA 20166 (US).

(72) Inventors; and

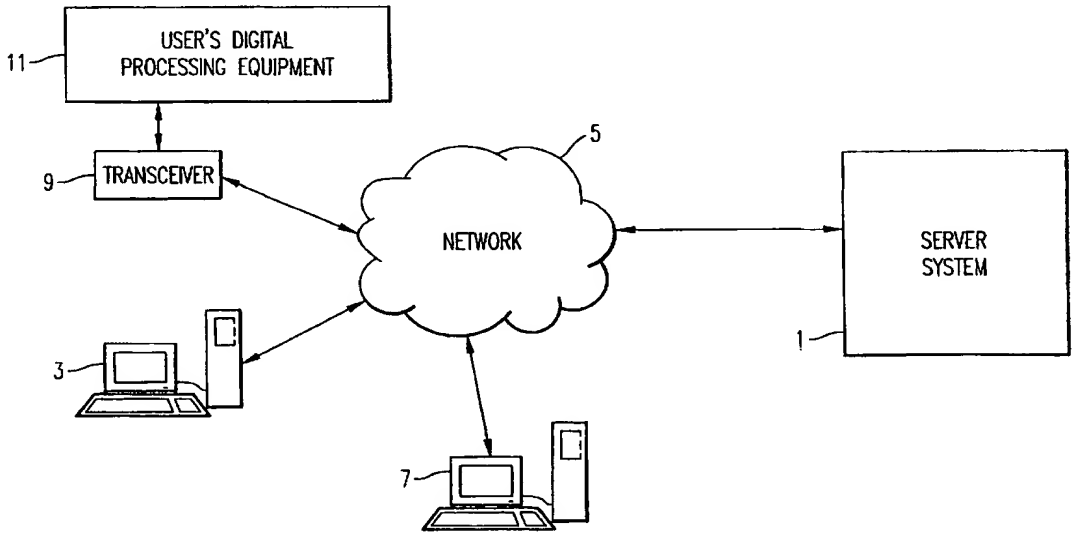
(75) Inventors/Applicants *(for US only)*: **SCHWARTZBARD, D., Aaron** [US/US]; 11015 Becontree Lake Dr. # 207, Reston, VA20190 (US). **ENNIS, Jeffrey, Taylor** [US/US]; 10310 Grandhaven Ave., Upper Marlboro, MD20772 (US). **SHINN, Michael, Thomas** [US/US]; 5513 Seqoia Farms Dr., Centreville, VA20120-3303 (US). **SHINN, Scott, Reid** [US/US]; 14401 Filly Ct., Centreville, VA20120-1639 (US). **CABACUNGAN, Robert, Spassky** [US/US]; 4327 Runabout Ln., Fairfax, VA22030 (US). **GALATIS, Stephen, George** [US/US]; 4153 Dawn Valley Ct. # 76E, Chantilly, VA20151 (US).

(74) Agent: **LYTLE, Bradley, D.**; Oblon, Spivak, McClelland, Maier & Neustadt, Crystal Square Five, Fourth Floor, 1755 Jefferson Davis Highway, Suite 400, Arlington, VA 22202 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ,

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR DATA AND MEDIA MANAGEMENT



(57) Abstract: A media management system, apparatus and computer program product employs a dynamic profile to suggest uniquely characterized pieces of content. The pieces of content are reliably identified, classified and affiliated with attributes associated with the respective pieces of content. A search method is employed to determine if the pieces of content have been cataloged. The dynamic profile is employed to facilitate managing media for use by a user of a client, and enable a client to select services offered by service providers who only view information about that user of the client as a "demographic unit". Media management system in a network environment includes client (11) and server (1) connected to the network (5) and a transceiver (9) to send and receive messages from the network.

WO 02/17146 A1

NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) **Designated States** *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

For two-letter codes and other abbreviations. refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Method and Apparatus for Data and Media Management.

CROSS-REFERENCES TO RELATED PATENT DOCUMENTS

This document claims priority to the following six commonly owned, co-pending provisional applications, of which numbers 1 through 5 were filed on January 7, 2000 and number 6 was filed on March 6, 2000, each of which are hereby incorporated by reference.

1. Ser. No. 60/174832, attorney docket No. 10638-0007-2 PROV, entitled "GRAPHICAL METHOD, SYSTEM, APPARATUS & COMPUTER PROGRAM PRODUCT FOR SKINNING GRAPHICAL USER INTERFACES,";
2. Ser. No. 60/174,835, attorney docket No. 10638-0008-2, entitled "SINGLE-STEP MULTIHOST QUERYING APPARATUS, METHOD AND COMPUTER PROGRAM PRODUCT"
3. Ser. No. 60/174831, attorney docket No. 10638-0009-2 PROV, entitled "REDUNDANT ARRAY OF INEXPENSIVE SERVERS APPARATUS, METHOD AND COMPUTER PROGRAM PRODUCT"
4. Ser. No. 60/174833, attorney docket No. 10638-00010-2 PROV, entitled "ENHANCED ADVERTISING FOR PROGRAMS BROADCAST TO COMPUTERS OVER A NETWORK"
5. Ser. No. 60/174842, attorney docket No. 10638-0011-2 PROV, entitled "METHOD, APPARATUS, AND COMPUTER PROGRAM PRODUCT FOR A MEDIA MANAGEMENT SYSTEM AND COMPONENTS THEREOF"
6. Ser. No. 60/187,030, attorney docket No. 10638-0025-2 PROV, entitled "METHOD, APPARATUS, AND COMPUTER PROGRAM PRODUCT FOR AN ADAPTIVE DATA STREAM VOLUME CONTROL", the entire contents of each of which, including any appendices, being incorporated herein by reference.

APPENDICES

The following appendices are attached hereto and should be construed as part of the present disclosure:

APPENDIX A: U.S. PATENT NO. 5,798.980

APPENDIX B: U.S. PATENT NO. 5,818,972

APPENDIX C: FIVE ARTICLES ON ASSOCIATION RULES

1. Mannila, H. et al, "Discovering Frequent Episodes in Sequences"
2. Lane, T., et al, "Sequence Matching and Learning in Anomaly Detection For Computer Security".
3. Agrawal, R., "Mining Sequential Patterns".
4. Ramakrishnan, S., et al, "Mining Generalized Association Rules".
5. Lee, W. et al, "Data Mining Approaches For Intrusion Detection".

APPENDIX D: Mount, D., "ANN Programming Manual", Version 0.1 (Beta released)

BACKGROUND OF THE INVENTION

Field of the Invention

The present invention is directed to methods, apparatuses, and computer programs that relate to computer-implemented data management for media and other data that include enhancements to evaluate or sense data by characterization or categorization for storage, power adjustment or other operation, enhancements to store or output the data that are scalable and reliable, enhancements to customize user interfaces, and enhancements to facilitate the delivery of advertisements to locations on a network. The methods, apparatuses, and computer programs of the present invention include features to ensure the anonymity of the user while facilitating the direction of services and programming to addressable users.

Discussion of the Background

The proliferation of computers, and a communication infrastructure for interlinking the computers has given rise to a set of tools that enable users to quickly and continuously receive data made available from source providers. These tools implement data formats such as REALAUDIO, WINDOWS Media Player, and MP3. By way of example, one type of data is audio data that is saved in MP3 (MPEG 3 audio) format and distributed from various source terminals to destination terminals. These destination terminals employ various

computer-based applications or programs, often described as players, that are configured to receive the files, buffer the files, decode the files, and "play" the files through an appropriate input/output (I/O) mechanism, such as an audio transducer, like a speaker. These players have also been used to broadcast programming over the Internet that resembles the content of conventional broadcast FM/AM radio stations. One type of conventional player is REALPLAYER version 6.0, 7.362 provided by Real Networks, Inc. Such players provide an "audio-on-demand communication system", like that described in U.S. Patent No. 5,793,980, a copy of which is provided as Appendix A and is incorporated herein by reference.

Other players are available for providing a visual display of a saved digital file as provided by way of a stream of video data. Various techniques are employed to provide moving video images over bandwidth-constrained communication channels for display by way of a video player at a user's terminal. One technique is described in U.S. Patent No. 5,818,972 provided herein as Appendix B and the entire contents of which being incorporated herein by reference.

Conventional players enable the decoding of streaming content, which are typically relatively large data files with respect to the channel capacity of the communications channel over which the content is sent. The content may be digital audio recordings, digital video, or other files to be distributed in a highly compressed form over a bandwidth-limited channel so as to enable an end user to have the impression that the user is receiving "broadcast" audio and video programming. Although the broadcast of radio and television programming over the Internet permits the programming to reach audiences that would not otherwise have access to the programing through conventional means (e.g., if the audience was out of the broadcast range of the station), there are many problems and disadvantages with such a system.

One of the features of such players is that the player enables users to select different sources of information, such as stations that broadcast content, or simply source sites that have saved thereon, a catalog of different content that a user may identify by way of the player for sending to the user's terminal. Each content is labeled with a conventional label such as a name of the piece of content (e.g., the title of a song, for example). Associated with the content may be other attributes associated with that content, such as the genre, artist, producer, etc. These labels are intended to be unique so that each content may be uniquely

identified, provided that the user knows what work the user is interested in receiving.

Figure 44 is a schematic diagram of a communications system that permits television or radio broadcasts to be transmitted over a computer network and played to users at their computers. A television or radio station 1101 transmits signals carrying the broadcast to a streaming server 1103. The streaming server 1103 converts the signal received from the station 1101 into a format that is readable by a computer. The formatted signal is sent to clients 1117, 1121, and 1125 over the Internet 1129. Thus, users at the clients 1117, 1121, and 1125 receive television and/or radio broadcasts over the Internet 1129 without having to own or operate a television or radio.

Labels associated with the content may also be used for searching purposes so that work that has been grouped-together by others as being of a similar genre, for example, may be identified by the user for future play-list inclusion. In order to ensure the integrity of the labeling information provided with the content, the present invention implements digital signatures may be employed to enable a player application hosted on the user's terminal to verify that a content has been provided in an un-altered format. The present inventors recognized that the labels applied to content are not optimally characterized by way of a digital signature because if even as much as one byte is incorrect in the original content, then the work may be completely mischaracterized. Such is often the case with communication links that may impart some high bit error rates, particularly when the communication links are wireless, analog communication lines. Furthermore, the digital signatures are not representative of the content itself, but rather only representative of the unique label applied by a person, without regard for the actual characteristics of the content. The integrity of the attributes and content is called into question since it has been observed that some users will unexplainably change the attributes associated with a particular piece of content. Thus, for cataloging purposes, it is often the case that content is mischaracterized in terms of attributes associated therewith.

As further identified by the present inventors, conventional player technology is deficient in numerous aspects, such as in violating a user's interest in remaining anonymous and not having personal information collected and used without the user's authorization. As a practical matter, in order to provide for a profitable business model for player-providers (i.e., companies that distribute players to subscribers), the player-providers obtain user-

sensitive information, often obtained surreptitiously, so that the player developers may develop a "profile" about that user for either sale to third parties, or for targeted advertising and the like by the player-providers.

Due to differences in recordings and differences in digital encoding techniques, digital music collections frequently contain songs having a range of volume levels. Furthermore, within a single song, volume levels may change drastically. In either case, a person listening to the music may need to continuously monitor and adjust volume levels to maintain consistency. An audio stream is represented in digital mechanisms as a series of numbers. These numbers are referred to as "samples". Each sample represents the magnitude of the audio stream waveform at some point in time. As presently recognized, the volume of audio data is directly proportional to the amplitude of the corresponding analog waveform data. Thus, volume adjustment would be ideally performed by normalizing the digital waveform before sending the digital waveform to an audio output device. Also presently recognized, normalization of the waveform is made difficult due to several properties of the audio data. First, audio data is "textured" meaning that audio data normally contains very irregular patterns of overlaid waveforms. The sounds produced from these irregular patterns of overlaid waveforms result from the changes in the waveform over time. In view of this operation the present invention recognizes that a single sample cannot be normalized in isolation; the sample should be normalized with respect to a context defined by other samples that are temporally close. Because an effective mechanism for dynamically changing the volume as appropriate on an "on-the-fly" basis, a normalization of the waveform should also be able to change. However, any instantaneous change will introduce a discontinuity into the waveform, and will be detectable by a listener as a crackle in the audio.

Even though Internet broadcasts increase the size of an audience, no extra revenue is generated from advertisers who pay for commercials to be aired out of the station. In other words, advertisers do not pay the stations (or the broadcast networks that own the stations) money specifically for the advertising that reaches the Internet audience. Also, just as in conventional broadcasting, the same advertisements are sent to every member of the audience, whether or not they are receiving the broadcast via the Internet or by conventional systems. As a result, advertisers whether through conventional broadcast or through the Internet cannot target advertisements to individual consumers and cannot send different

advertisements to different consumers. The present inventors recognized this weakness and provide the present invention to generate extra revenue from advertisers for commercials that are delivered over the Internet.

Another aspect of the system that the present inventors recognized was that as the amount of information being maintained in an electronic format increases, the ability to make that information readily available to all who require access to it becomes an increasingly difficult challenge. The challenges posed by this problem are both technical and financial in nature. One approach is to scale vertically by buying more expensive, more capable server machines. The present inventor has recognized that scaling vertically can be prohibitively expensive, and results in an architecture where the server is a single point of failure for the entire system. A less expensive approach is to scale the server system horizontally. One popular technique for scaling horizontally is to employ a redundant array of inexpensive disks (RAID). An advantage to the RAID approach is that since the information is stored redundantly, the availability of the information is more reliable. However, the present inventor has recognized that these systems are expensive to configure and that the system maintaining the RAID array is a single point of failure for the system. Replacing this server machines with more capable hardware is an expensive alternative, and does not address the concern that the server is a single point of failure in the system. Replacing the single server with multiple servers, while addressing the single point of failure problem, and being relatively inexpensive, raises new challenges. For example, queries in a system that used to be routed to a single server must now be routed to the appropriate server. Additionally, the information being maintained at the server must either be replicated across all of the servers or alternatively, a partitioning scheme must be devised that will cause added complexities to maintaining the information repository. Furthermore, the inventor recognized that the routing and data partitioning issues raised by the multiple server architecture can make that alternative prohibitive. The challenges, then, as presently recognized, are to develop an approach to provide access to large amounts of information in a manner that is acceptable to the user, is scalable with respect to capacity, is reliable, and has acceptable performance and to provide a mechanism through which multiple inexpensive servers can be queried with a single query and only that server, or those servers, that can satisfy the query will respond to the query.

With the proliferation of personal computers in the work place and in the home has come the diversification of the personal computer user population. With this diversification has come the need for the development of tools and techniques to empower those personal computer users without an advanced computer literacy. These tools and techniques include a graphical user interface provides an intuitive way for a user to interact with a software application. The graphical user interface may include a skin that is a set of images provided by the user to define a custom look and feel of an application. A skinable software application is one that can have its default look and feel replaced by a custom skin. To skin the graphical user interface (or so-called skinning of a software application) consists of providing the custom images defining the look and feel of the application and mapping the software application's functionality to the controls as defined by the user in the images making up the skin. Stated simply, skinning an application involves replacing those images that the software application presents to the user with a set of images selected by the user. Skinning a software application is a technique that has been used by those with an advanced computer literacy for sometime. The present inventor has recognized that the skinning techniques used to date are tedious in nature and require an understanding of the underlying technical concepts such as pointing devices and the interaction between those devices and a software application itself. Given that skinning a software application provides, for the most part, purely aesthetic benefits, these techniques have failed to gain popularity due to their time consuming and tedious nature. The challenge, then, is to provide a skinning technique to a growing and diverse user population in a manner that will be acceptable to a broad cross section of this user population.

To invoke the functionality of the skinning software application, the user simply places a pointing device (e.g., a cursor as driven by a mouse) over an icon on the display corresponding to an action to be invoked by the software application, and then selects the action with the pointing device (e.g., clicking the mouse button). To enable this custom user interface, the user maps regions of the custom images to the functions to be performed by the software application. As an example, a user may want a region on his custom image depicting a circle to correspond to the same function that is invoked in the software application by the default graphical user interface by selecting a gray rectangular region identified by the literal "OK". To define this circular region to the software application, the

region is defined in terms of its x and y pixel position coordinates. This definition process is very time consuming and tedious in that the pixel regions are described in a text file by typing pixel locations and dimensions of the regions to be defined. The present inventor recognized that an inherent flaw of this technique is that the regions are defined rectangularly. This limits the flexibility that the user has in defining the regions of the custom interface. Moreover, the present inventor recognized that the current techniques are limited in the flexibility they allow with respect to the placement of the regions on the image.

Figure 49 illustrates the complexities and limitations of the prior art techniques for skinning software applications. Figure 49 shows an image that a user will define to the software application as a replacement graphical user interface, or skin for a software application. The image that the user has selected has a rounded rectangular shape 2000. It has three regions that the user will define to the software application as corresponding to three functions provided by the application. Those regions are shown in Figure 49 as a circle 2002, a rounded rectangle 2004, and a rectangle 2006. If, as an example, the software application was a media player, the circular region 2002 might correspond to a stop function, the rounded rectangular region 2004 might correspond to a play function, and the rectangular region 2006 might correspond to a pause function.

To skin the media player application, the user defines the regions corresponding to the functions provided by the player to the application. In the prior art, this would be accomplished by creating a text file and defining those regions in the text file. As shown in Figure 49, one of the limitations of the prior art is that the regions that correspond to the user interface in general, and the controls in particular, are defined in terms of rectangles and not, in terms of the shapes of the regions that the user has decided on. To define the image shown in Figure 49 as a skin for a media player application the user would define four regions and three functions to the application. In particular, the user would identify the overall general shape of the image 2001, the shape corresponding the stop function 2003, the shape corresponding to the pause function 2006, and the shape corresponding to the play function 2005.

In order for a user to define a shape to the application under the prior art, the user would determine, as an example, which pixel corresponds to the upper left corner of the area being defined, and then the horizontal and vertical pixel counts defining the rectangular

region of the area. As shown in Figure 49, for the area corresponding to the play region 2005, the user would specify in text format, the (x, y) coordinate corresponding to the upper left corner of the play region. In this example, the user would specify (17, 40) as the upper left corner of the play region. The user would then specify that the region is defined by the rectangle with an upper left coordinate of (17, 40), a height extending 10 pixels to coordinate (17, 50) and a width of 25 pixels extending to coordinate (42, 40). Clearly, specifying all the coordinates for all of the controls of a complex user interface would be an extremely tedious and time consuming process. Moreover, any changes made to the overall image 2001 may cause a shifting of the coordinates that have been used to define the regions corresponding to the functions of the application. If this were the case, the user would modify the descriptions so that the coordinate definitions of the areas were accurate.

SUMMARY OF THE INVENTION

Consistent with the title of this section, a brief discussion of aspects of the invention will now be highlighted. However, this section is not intended to replace the more comprehensive explanation and description of the invention, which is provided by this entire document.

An object of the present invention is to overcome the above-identified and other limitations associated with conventional systems, apparatuses, methods and computer program product, associated with presenting, distributing and searching through digital data that is of interest to a user.

To this end, the present invention provides a system and method for managing media through a set of mechanism distributed between a client and a server system. The client, or server, is configured to characterize a piece of content (e.g., a data file that contains information, such as an audio file, video file or more generally a stored file representing measurements of context-related discrete events) with an efficient cataloging technique.

In an audio player environment, an analog signature is developed for a particular piece of content where the analog signature is derived from the content itself, and not arbitrarily assigned to the content without regard to the characteristics of the content. Having characterized different pieces of content with unique analog signatures, enables the server system to efficiently store the analog signatures in an N dimensional space. Using the

efficient storing approach enables fast and efficient searching for the analog signatures when determining whether that particular content has already been stored in the data structure.

Another feature of the present invention is that once stored, attributes associated with a particular content are permanently and reliably attached to that content. Moreover the content and the attribute information are "locked" to one another so regardless of the communications channel (e.g., a communication link or a memory), the content may be transferred between many users but the integrity of the attributes and the content is reliably maintained.

Another feature of the present invention is that users of the client application (which may include a player, or some other mechanism for managing media) communicate to the server in an anonymous fashion so that the "real identity" of the users are not distributed or even known to the server or others who would like to communicate with the users, such as advertisers. A dynamic profile is developed for a particular user so that a temporal pattern of activity by the user is developed to form a baseline set of discrete events associated with a particular user. This baseline set of discrete events may be used to support "agent" software processes that are employed to automatically provide services to a user without the user having to make a specific request for the service.

The present invention also includes features that enable a user to customize the user interface according to the likes or dislikes of that particular user. Moreover, a skinning system is employed to quickly facilitate a user being able to customize the graphic user interface according to the tastes of that particular user. The present system also includes provisions for providing a high integrity system at the server that would avoid system shut-down if one of the servers of the server system happens to fail or have a problem associated with components of that particular server.

The present invention also includes provisions for using the dynamic profiles in several different ways, including profile sharing, predictive advertisement, creation of presentation of "ECONS", and profile-influenced audio advertisements.

Another object of the present invention is to provide a method and apparatus for automated monitoring and adjustment of volume levels of a digital audio system. A feature of this automated monitoring and adjustment of volume will be to provide a mechanism by which adaptive normalization of the waveform is accomplished between adjacent sections of

audio data (e.g., within a song or between adjacent songs).

Accordingly, one object of this invention is to provide a novel method, system, and software product that allows broadcast networks and/or broadcast stations to generate extra revenue from advertisers for commercials that are delivered over the Internet.

Another object of the present invention is to provide a novel method, system, and software product that provides advertisers and broadcast networks flexibility in the purchase and sale of commercial time during Internet broadcasts.

It is a still further object of the present invention to provide a novel method, system, and software product that allows advertisers to deliver targeted advertisements to an Internet audience, based on demographic information of individual audience members.

It is still another object of the present invention to provide a novel method, system, and software product for delivering and receiving commercials during an Internet broadcast and that does not require, and/or is compatible with, standard Web browser software.

It is an even further object of the present invention to provide a novel method, system, and software product for displaying a live link to a Web site associated with an advertiser while that advertiser's commercial is being played and to display the link in the form of a logo corresponding to the advertised product or service.

It is yet another object of the present invention to provide a novel method, system, and software product that enhances the value of advertisements broadcast over the Internet while providing little or no additional cost to the broadcast network or station from which the broadcast originates.

These and other objects are achieved according to the present invention by providing a novel method, system, and software product for delivering commercials to a computer. A broadcast signal is received from a broadcast station. The broadcast signal includes commercial segments when broadcast commercials are broadcast and program segments when no broadcast commercials are broadcast. The broadcast signal is converted into a first data signal, which is delivered over a computer network to a computer during the program segments of the broadcast signal. A second data signal, corresponding to computer network commercials that are at least partially different from the broadcast commercials, is generated and delivered over the computer network to the computer during the commercial segments of the broadcast signal. As a result, the computer receives the computer network commercials

instead of the broadcast commercials during the commercial segments of the broadcast signal.

In this manner, an audience receiving the broadcast over a computer network, such as the Internet, receives a different set of commercials (i.e., the computer network commercials) than the audience receiving the commercials broadcast over the air. Thus, the sale and purchase of commercial time for the computer network broadcast can be separated from the sale and purchase of commercial time during the standard network broadcast. Since all of the processing may be performed outside of the broadcast station, there is no need for the broadcast station to make expenditures to implement the present invention.

In a preferred embodiment of the invention, users are assigned to user groups based on their user profiles. The computer broadcast commercials are associated with the user groups, and each computer is associated with the user group of the user at the computer. Since different commercials may be associated with different user groups, each user group receives different commercials targeted to its users. Preferably, each computer receiving the broadcast is associated with a user group determined on the basis of a user profile of a user. The user profile preferably includes demographic information such as age, gender, zip code, profession, and income. The present invention also enables tracking of users' event logging information, including the length of time that users listen to various broadcasts and commercials.

The present invention also provides a novel method, system, and software product for playing network commercials on a computer. A first data signal, generated from a broadcast signal sent from a broadcast station, is received at the computer. The first data signal includes commercial segments when broadcast commercials are broadcast and program segments when a program is broadcast. The program is reconstructed from the first data signal and played on the computer during the program segments of the broadcast signal. The computer receives a second data signal, carrying computer network commercials, during the commercial segments of the broadcast signal. The computer network commercials are reconstructed from the second data signal and played on the computer during the commercial segments of the broadcast signal. As a result, the computer plays the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

In a preferred embodiment, a screen object associated with an advertiser is displayed

while the advertiser's commercial is played on the computer. The screen object is preferably an icon or icon-like image in the shape of a logo corresponding to the service or product being advertised. The screen objects may also provide active links to network sites corresponding to advertisers. This advantageously permits users to connect their computer to a manufacturer or product sites while that manufacturer or product is being advertised. This feature also serves to increase recognition of a company's trademark as well as to keep users occupied during the commercial segments of the broadcast signal.

The present inventor recognized the limitations in the approaches currently being used to design scalable and reliable server systems. Accordingly, the object of the present invention is to provide a server system that is cost effective, is scalable with respect to storage capacity needs, is sufficiently redundant so as to eliminate the single point of failure problems with conventional server systems, is not overly complex from a user perspective, and that has acceptable performance characteristics.

To address these concerns, features of the invention include a computer based system, method, and computer program product, that presents a simple file system interface to a redundant array of inexpensive servers. By replicating the entire server, the present invention provides redundancy in all aspects of the server system.

In one implementation, the present invention abstracts a file system that redundantly maintains information on multiple server machines. A database is maintained in the system to manage the abstraction of the file system. The users of the file system are presented the abstract view of the file system so as to hide the underlying complexities that result from redundant storage and performance optimization. The system includes software that will detect server failures and implement recovery by redistributing the load of the failed server without interruption of the system.

The present inventor recognized that by providing a server system based on multiple inexpensive servers, a cost effective, reliable server system can be designed where neither the processors nor the storage devices are single points of failure in the system. The present inventor has developed a reliable, self-adjusting system with built-in redundancy and failure detection and recovery. Furthermore, the inventor recognized that the complexities of the server system can be hidden from the user by abstracting the server file system so that the logical view of the file system presented to the user does not reflect the physical complexities

of the redundant storage system.

The present inventor recognized the limitations in the approaches currently being used to deal with server performance issues and server single point of failure concerns with typical client-server architectures. Accordingly, an object of the present invention is to provide a mechanism through which multiple inexpensive servers can be issued a single query from a client, and only that server, or those servers, that are capable of fulfilling the query will respond to the query.

To address these concerns, features of the invention include a computer based system, method, and computer program product, by which broadcasting and multicasting built-in Internet Protocol (IP) broadcasting protocols are used to query the servers. By having clusters of server machines configured to react to multicast or broadcast messages, an intermediate layer of a server architecture can decide which cluster should receive a particular query, format the query according to the configuration of the cluster to be queried, then multicast (or broadcast) the query to the appropriate server cluster. A more complete description of the IP protocols, and the broadcasting protocols in particular is provided in the literature, for example in Stevens, W. Richard, "TCP/IP Illustrated, Volume 1: The Protocols," Addison-Wesley Publishing Co., 1994, ISBN 0201633469; pp. 169-178, the contents of this book being incorporated herein by reference.

Thus, the present invention does not require the purchase of expensive hardware, and it addresses both the routing and the data partitioning issues experienced with multiple server architectures. The present inventor recognized that by sending every server a request, that there would be no routing issues. The inventor also recognized that since only those servers that can answer the request or respond to it, there would be no data partitioning issues. The information held on each server can be arbitrarily determined.

The inventor of the present invention recognized the limitations in the current software application skinning techniques. Accordingly, the object of the present invention is to provide graphical methods, systems, apparatuses, and supporting tools for generating user customized skins for software applications that is non-technical in nature and which is easy to employ.

The present inventor recognized that a text file containing rectangular approximations of regions of an image should not be required to used that image as in element of a software

application skin. Accordingly, one object of the present invention is to provide a solution to this problem, as recognized by the present inventor so that a user would be able to define any region, regardless of its shape, to the software application. Furthermore, the present inventor recognized that a user should not be required to define that region of the image in a textual manner. Accordingly, a further object of the present invention is to provide a solution to this problem, as recognized by the present inventor.

The present inventor recognized that the pixels of a displayed image contain more information than just their (x, y) positional information. In particular, the present inventor recognized that a pixel's color information, as determined by its corresponding red, green, and blue settings, is a property that could be easily used to define regions of interest of an image. Furthermore, the inventor recognized that using predetermined color values on a mask overlay to correspond to regions of interest would not have the inherent limitations that the regions must be rectangular, nor would resizing the image pose a problem since the color coded regions would resize in accordance with image resizing.

To achieve these objects, the present inventor has invented a novel computer based system, method and computer program product, by which color coded mask overlays can be created and compiled such that the regions of interest on the user's skin image are defined to the software application. Using a graphical, color based technique allows any computer user with a minimal understanding of simple drawing application such as COREL DRAW or ADOBE PHOTOSHOP to create their own skins for software applications that can be skinned.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete appreciation of the invention and many of the attendant advantages thereof will be readily obtained as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings, wherein:

Figure 1 is a block diagram of a system-level description according to the present invention;

Figure 2 is a block diagram of a client terminal according to the present invention;

Figure 3 is a block diagram of a redundant array of independent servers according to

the present invention;

Figure 4 is a flowchart of a process for how analog signatures are identified and conveyed according to the present invention;

Figure 5 is a flowchart of a process for developing an analog signature for a particular piece of content;

Figure 6 is a flowchart of a process for identifying whether an analog signature is already present within a data structure, and associating that analog signature with attributes of that piece of content, if it exists;

Figure 7 is an exemplary data structure showing fields of a message transmitted over a communication network, where the message holds an analog signature of a particular piece of content;

Figure 8 is a frequency diagram showing a spacing of respective frequency bands employed when characterizing a content and developing an analog signature;

Figure 9 is a flowchart of a conventional approach for providing a digital signature of a data file;

Figure 10 is a flowchart of a process according to the present invention, that provides a header-locking mechanism so as to ensure a message content, or information associated therewith, is not tampered with when received at a destination terminal;

Figure 11 is a flowchart of a process showing how association rules are employed to develop a dynamic profile according to the present invention;

Figure 12 is a flowchart of a process for implementing a dynamic profile mechanism;

Figure 13 is a flowchart of a process for extracting transaction data from a transaction table, identifying association rules from the extracted data, and normalizing the probability of the different events that make up the association rules;

Figure 14 is a graphical representation of how a hash table helps to enable an efficient collection of data for efficiently creating association rules;

Figures 15a-15c show components used to build a directed graph used to normalize probabilities of association rules;

Figure 16 is a Venn diagram used to illustrate why a normalization of association rules helps to avoid a convergence of a dynamic profile constructed from unnormalized association rules;

Figure 17 is a block diagram of a fixed system for managing media according to the present invention;

Figure 18 is a block diagram of a mobile client based system similar to the system of Figure 17;

Figure 19 is a block diagram of a profile sharing and editing system according to the present invention;

Figure 20 is a two-dimensional embodiment of an N dimensional signature space according to the present invention;

Figure 21 is a two-dimensional signature space that is divided in one dimension;

Figure 22 is the two-dimensional signature space of Figure 21 that is further divided in another dimension;

Figure 23 is a tree diagram used to track neighbor regions;

Figure 24 is a data structure of information saved for a node of the tree in Figure 23;

Figure 25 is a flowchart of a process for determining whether a neighbor status exists between two leaf nodes;

Figure 26 is a flowchart of a process for efficiently searching an N dimensional signature space;

Figure 27 is a flowchart of a process for dividing the N dimensional signature space as part of an efficient search process;

Figure 28 is an exemplary graph of a waveform showing adjacent sections of the waveform, along with a scale factor;

Figure 29 is an exemplary graph of a waveform like that shown in Figure 28, although showing that an instantaneous volume level of an adjacent section of the waveform may exceed a predetermined level; and

Figure 30 is another graph of a waveform showing a scale factor that has been adjusted to account for a waveform like that shown in Figure 29.

Figure 31 is a schematic diagram of a communications network used to deliver commercials to the audience of an Internet broadcast in accordance with the present invention;

Figure 32 is a password table for associating each username with a corresponding password;

Figure 33 is a group identification table for associating each username with a group to which the corresponding user has been assigned;

Figure 34 is a user profile table for associating each username with demographic information of the corresponding user;

Figure 35 is an event logging table for associating each username with event logging information of the corresponding user;

Figure 36 is an advertisement list table for associating each group with a play list of advertisement identifiers (ad IDs);

Figure 37 is an advertisement file table for associating each ad ID with a memory location where the corresponding computer network commercial is stored;

Figure 38 is a screen objects table for associating each advertisement identifier with the memory location of one or more screen objects;

Figure 39 is a flowchart of a process for registering a user with an Internet broadcast service;

Figure 40 is a flowchart of a process for delivering a broadcast to a computer over the Internet;

Figure 41 is a flowchart of a process for delivering targeted advertisements to a computer during an Internet broadcast;

Figure 42 is a graphical user interface, including screen objects associated with commercials and advertisers;

Figure 43 is a schematic illustration of a general purpose computer programmed according to the teachings of the present invention; and

Figure 44 is a schematic illustration of a conventional system for providing Internet broadcasts of television and radio programming.

Figure 45 is a block diagram showing a system configuration of an embodiment of the present invention; and

Figure 46 is a diagram showing an abstraction of a file system according to the present invention.

Figure 47 is a block diagram showing a system configuration of an embodiment of the present invention; and

Figure 48 is a flow diagram of a method performed by the system as shown in Figure

47 according to the present invention.

Figure 49 shows a user image that will be used as a skin for a software player application and which illustrates the complexities and limitations of the conventional techniques used to create skins for software applications;

Figure 50 shows a user image that will be used as a skin for a software player application with the controls in their default states;

Figure 51 is the image of Figure 50 with the controls shown in their selected states:

Figure 52 shows a color coded mask overlay wherein the color coded regions correspond to the regions defining the controls in Figures 50 and 51;

Figure 53 shows a menu animation mask wherein the color coded pairs of regions define the inactive and active positions for icon making up a menu system;

Figure 54 shows an image of all displayable characters for a font to be used to present text on the skin;

Figure 55 shows a mask capturing the horizontal spacing required for each of the displayable characters shown in Figure 54, allowing the skin to display text as a true-type scalable font;

Figure 56 shows an information block that is generated by the byte code compiler as a result of compiling the images making up the software application skin;

Figure 57 is a flow diagram of a method performed to create a skin for use as a graphical user interface for a skinable software application according to the present invention; and,

Figure 58 is a flow diagram of a method performed by the execution of a software application that has been skinned according to the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

MEDIA MANAGEMENT SYSTEM

Referring now to the drawings, and more particularly the system-level block diagram of Figure 1, Figure 1 shows a network-based implementation of a media management system according to the present invention. A server system 1, which will be described in more detail with regard to Figure 3, connects to a network 5. The network 5 includes the Internet, although other networks, including proprietary networks, or publicly-available networks are alternatives as well. Connected to the network 5, are a client 3, another client 7, and digital processing equipment 11, by way of a transceiver 9. The digital processing equipment 11, is similar in operation to the client 3, although not necessarily a traditional personal computer, as is shown by the client of Figure 3. In particular, the digital processing equipment 11 and transceiver 9, may be a wireless telephone, wired or wireless general purpose communication device, such as a palm computer offered by Three-Com, a RIO player or the like. In each case, the client 3, another client 7, and digital processing equipment 11, provide a mechanism by which a user may communicate with the server system 1 by way of the network 5. It is also possible, however, for a communication session to be implemented directly to the server system 1, or even between each other. Alternatively, the several clients need not be electrically (wired or wireless) connected with the server system 1, but may also communicate by exchanging computer readable media, such as a CD ROM, or other mechanism by which data files are stored.

For convenience, the principle operation of the system shown in Figure 1 will be explained via communication between the client 3 and the server system 1 by way of the network 5. However, it should also be recognized that any number of other clients, such as those illustrated in another client 7 and digital processing equipment 11 may also be employed in communicating with the server system 1, and also being interconnected by way of the network 5 to each other.

The client 3 a software-based application that processes and presents "content". "Content" in this instance is a particular data file that may represent a digital audio recording, video image or other data file (such as text-based media) that may be of interest in viewing or listening-to by the user of the client 3. Additionally, the "content" may also be a particular

service that is ordered by the user, used by the user, or perceived as being usable by the user. In the case of the media being a service, the embodiment is directed to an "agent" application where different services are predicted as being useful to the user and present to the user for selection by the user, if the user opts to select the service.

In the present embodiment the client 3 employs a player, that is an audio player for presenting in audible format data files that either reside on the client 3, or are received by way of the network 5 from any one of the other clients, or the server system 1. Alternatively the player is an audio/video player or simply a video player.

In a typical operation the client 3 retrieves from memory, a piece of content that may have been encoded by the server system 1. Different types of readers employ various decoders that decode audio files for example, MP3, REAL AUDIO, LIQUID AUDIO, WINDOWS MEDIA, or the like. The client 3 includes a CODEC (coder/decoder), or simply a decoder, that is appropriate for the encoding technique applied to the data. Audio, video and other information that has highly redundant content is compressed by encoding prior to being sent over the network 5 so as to minimize the amount of time and channel resources required to distribute the information over a bandwidth-limited channel, such as the Internet.

Figure 2 is a block diagram of separate components within the client 3. As seen, the client 3 may be a personal computer that employs one or more processors 205 that communicate with other mechanisms by way of a bus 203. While the present description identifies the different subcomponents within the client 3 in the form of separate mechanisms, it should be recognized that the different mechanisms may be performed by way of software instructions executed on the processor 205. Alternatively, the mechanisms may also be partially performed in software and partially in hardware such as by way of specialized components like a programmable array logic (PAL), or application specific integrated circuit (ASIC).

An input/output unit 213 communicates with the network 5 by way of a communication port 201. The communication port 201 may be a coaxial cable in the case of the input/output unit 213 implementing a cable modem connection, or a RJ-11 connection when a dial-up functionality is included in the input/output unit 213. The input/output unit 213 also connects with sensors 211, which may include bio-sensors such as eye-movement tracking sensors, temperature sensors, tactile transducers, and other sensors that may be used

to identify a particular user-unique event, such as a heart rate, an applied force on a pointing object or the like. Input from the sensors 211 may be employed as attributes of discrete events used to form transaction data when developing a dynamic profile kept by the client 3 on that particular user. An external storage device, such as a hard drive, or external removable floppy drive, for example, is used to provide a local source of data, such as a mechanism by which a profile for another user may be inserted into the client 3. A local information source/sink 207 is also connected to the input/output unit 213. One example of the local information source/sink is a digitizing microphone that inputs digital data into the input/output unit 213 for recording and encoding by the processor 205. Likewise, the local information source/sink 207 may also be a recording mechanism by which audio files, in an encoded format may be stored thereon.

The input/output unit 213 connects to the bus 203, for exchanging information between the different mechanisms shown. The user interface 239 connects to the video/audio/sensory input/output and skinning mechanism 237. The user interface 239 may include a pointing device, such as a mouse, as well as display mechanism and speakers for presenting audio information and video information to the user. The skinning mechanism, as will be discussed, relates to providing a mechanism by which a user may customize the graphical user interface. The sensory aspect of the input/output mechanism 237, communicates with the input/output unit 213, and the information provided by the sensors 211, to influence the temporal aspects of the dynamic profile kept in the storage device 235. Moreover, the storage device 235, which in one embodiment is a semiconductor memory, holds thereon a profile 231.

The profile 231, as will be discussed in more detail herein, is a user-unique association of discrete events that are used to predict future events, based on a user's present and past activities. For example, in the audio player embodiment, the profile 231 includes an organized set of information regarding the audio files and an association between audio files that are associated with the user's listening habits, and other events performed by the user, or environmental events regarding a state in which the files were used by the user.

The profile application mechanism 229 uses the profile for deriving information therefrom so as to describe the user in terms of demographic unit. The profile application mechanism 229 also includes security, editing and profile sharing features as will be

discussed with respect to Figures 17-20, for example.

The byte code compiler 225 coordinates with the skinning mechanism 237 so as to provide a convenient manner in which items are to be displayed on user interface 239, according to the tastes of the user. Memory 227 connects to the bus 203 for use by the processor 205 in storing intermediate results, serving as a buffer, and also holding software instructions for the different mechanisms that are included within the client 3. The profile application mechanism 229, also includes features of combining profiles that are shared between the user of the client 3, as well as other profiles that were requested by the user by way of the local information source/sink 207. Alternatively, the user of the client 3 may request a profile from other persons, such as a celebrity, so that the user of client 3 may have listening selections offered to that user, and a play list that reflects that particular celebrity's preferences, for example.

Furthermore, the profile application mechanism 229, may also provide an opportunity for individuals to form special interest groups where users share profiles with one another so that the people can learn more about each other. One embodiment is similar to instant messaging, where in addition to sending messages among one another, profiles may be shared quickly amongst the different users so the others will learn more about each other's likes and dislikes with regard to particular content. The profile application mechanism 229 also provides the ability for the user to remain anonymous, while offering information to the server system 1, by way of the network 5, based on the optional information that the user chooses to provide to the server system 1. Moreover, the server system 1 may request that the user of client 3 provide to the server system 1, information regarding that particular user's income, education level, shopping preferences, and the like. However, the user need never provide any information regarding an actual identity of the user. In this way, the user is able to preserve the absolute anonymity of his or her real identification, yet still provide an "ethical" way to enable the server system 1 to develop a profile on that particular user. The user maintains the opportunity to alter the profile or even delete the profile that has been developed and saved by that particular user. Thus, once again, if the user is not satisfied with the information that is being distributed to other parties, by way of the server system 1, the user may choose to delete, alter, or modify the contents of the profile by way of the profile application mechanism 229.

The analog signature characterization mechanism 215 will be discussed below, as will each of the other mechanisms, and provides an ability by which an analog signature will be developed for a particular content that is of interest to the user of the client 3. Developing an analog signature, which is representative of the actual information contained as part of the content, rather than an arbitrary label, provides an ability by which associations and proximities of one content with respect to other contents may be obtained. Furthermore, the analog signature provides a much more reliable mechanism for determining whether a content being listened to (viewed) by one user is in fact a same content as a previously saved content, for which attributes are known by the server system 1.

The dynamic profiling mechanism 217 cooperates with the profile application mechanism 229 and the profile 231 of the storage device 235. As will be discussed below, developing a dynamic profile on a user, enables the user to have an agent recommend for that user, particular play lists or other content, based on past patterns of user activity. These patterns of activity may be influenced by temporal factors, as well as sensory factors.

Another mechanism included in the client 3, is a header locking mechanism 219, which provides a way for maintaining the integrity of content, as well as attributes of that content, after the attributes have been reliably linked with a particular content. Once the content and the attributes have been linked to one another, third parties cannot tamper with the attributes of the content, without the tampering being easily detected by the client 3, or the server system 1.

An ECON control mechanism 221, provides a way in which sponsors of the server system 1, may present links for targeted advertising of services to the client 3. Moreover, much like the dynamic profile is able to suggest future content, the ECONs provide a convenient way for service providers to offer their services based on the service provider characterizing that particular user as a "demographic unit".

An item set chooser mechanism 223 is described below, especially with respect to Figures 15a-15c and 16.

As shown in Figure 3, the system according to the present invention includes one or more clone computers 4000 which serve as an interface point to the system for client computers 4004 through link L4007 (also referred to as network 5 in Figures 1 and 2) to gain access to information maintained on the server system 4020. Each individual clone computer

(e.g., 4005, 4006) is configured identically. Therefore, the clone computers 4000 component of the server system is scalable to meet performance requirements, and reliable since the failure of any single cloned computer will not cause the system as a whole to fail.

The clone computers 4000 are connected through a switch 4001 to a database 4002 and the server machines 4003. Each individual clone computer (e.g., 4005) maintains its own connection with the switch 4001, although Figure 3 shows only one exemplary link L4007. Both the database component 4002 and the server machines component 4003 include redundant elements thereby providing the performance and reliability that is an objective of the present invention. For example, the information maintained in database 4011 will be maintained in parallel in database 4012. Each individual database (e.g., 4011, and 4012) will maintain its own connection with the switch 4001, although Figure 3 shows only one exemplary link L4010. The server machines 4003 component of the system includes multiple inexpensive servers (e.g., 4013) each of which will maintain a connection with the switch 4001, although Figure 3 shows only one exemplary link L4009. Each individual server machine (e.g., 4013) is configured to have one or more storage devices (e.g, 4016) connected to it. As will discussed below, the information being maintained in the server system of the present invention is stored redundantly such that the failure of any individual storage device, or any individual server machine does not preclude the access of the information held on that device or that server elsewhere in the system.

The invention includes a software monitoring agent 4017 that is implemented as a process running on each clone computer (e.g, 4005), each database (e.g., 4011), and each server machine (e.g, 4013), although other software architectures may be used as well. The software monitoring agent 4017 will detect and react to failures of components of the server system. When a component of the system fails, the software monitoring agent 4017 will cause the responsibilities that were maintained by that failed component to be redistributed to operational components of the server system. In this way, the server system of the present invention is a reliable and self-adjusting system with built-in failure detection and recovery capability.

The file load monitor 4018 is a software component of the server system that will report on which files and/or systems are being accessed within the server system most frequently. The file load monitor software 4018 will include the capability to make decisions

regarding the redistribution of those files most frequently accessed thereby maintaining a consistent loading throughout all components of the server system. Again, any adjustments made to the system by the file load monitor software 4018 must be reflected in the database 4002. Moreover, the activities of the file load monitor software 4018 allow the server system of the present invention to be seamlessly scaled by adding additional inexpensive servers to the architecture. The file load monitor software 4018 will take advantage of any available resources in the server system to distribute the load.

When a client computer 4004 requests a file from the file server system 4020, that request is made to a clone computer (e.g., 4005). In the present example, the client computer will communicate with the clone computer shown as 4005 in Figure 3, and individual database 4011 will be accessed, but it should be recognized that the invention does not restrict which individual client computer 4004 can communicate with which individual clone computer 4000 or which individual database 4002 is accessed. The clone computer 4005 will then query the database (e.g., 4011) using the logical user-determined name for the file requested by the client. The database 4011 will return to the clone a list of the physical locations and file names corresponding to the locations where the requested file is stored within the server system 4020.

The clone computers 4000 each maintain a route table (e.g., 4019) that cross-references the device addresses corresponding to the physical locations of the file to the machine address of the server machine on which that device is mounted. Through the use of the route table (e.g., 4019), then, the clone computers 4000 can determine based on the information received from the database which physical server machines 4003 are maintaining the requested file. By attaching to one of those server machines (e.g., 4013) containing the desired file, the clone computer 4000 can then access the requested file on the device (e.g., 4016) that is mounted on that server machine (e.g., 4013).

If the software monitoring agent 4017 detects the failure of a server machine 4003 it will cause the responsibilities of that machine to be redistributed within the system. For example, if a server machine were to be detected as inoperable, a "hot spare" (e.g., 4015) server machine may be instructed by the software monitoring agent 4017 to replace that failed server machine. By accessing the database 4002, the "hot spare" server machine (e.g., 4015) can determine which files were the responsibility of the failed server. The "hot spare"

server machine (e.g., 4015) would be required to obtain copies of all those files, and store them on devices that were mounted to that "hot spare" server machine. Once all of the required files were stored on storage devices mounted on it, the "hot spare" server machine (e.g., 4015) must communicate to the clone computers 4000 that files being reported by the database as existing on the failed server machine storage devices, are now to be found on storage devices mounted to the "hot spare" server machine. The clone computers 4000 will then update their respective route tables (e.g., 4019) to reflect this change.

If the server system does not have a "hot spare" server machine available (e.g., 4015), the results of the above-described failure detection and recovery can be obtained through an alternative approach. If the software monitoring agent 4017 detects the failure of a server machine, and no "hot spare" server machine is available in the system, the software monitoring agent 4017 can cause the load of that failed server machine to be absorbed by the operational components of the server system. To achieve this, the software monitoring agent 4017 will be required to interact with the database 4002 to determine the load from the failed server machine that must be distributed, and must update the database to reflect the redistribution. In this example, it would be unnecessary to update the route tables (e.g., 4019) of the clone computers 4000.

ANALOG SIGNATURE

In order to reliably associate attributes of a particular piece of content (i.e., "content") with the content itself, a reliable mechanism is needed to uniquely identify a particular content, even if that content has been slightly corrupted, perhaps due to channel disturbances. However, the task of recognizing a digitally encoded audio stream is formidable, and impractical for large audio collections, or collections of other content. Particular content, which may be a song in the case of an audio reader embodiment of the invention, can be performed either on the digital version of the song, or on the decoded analog version. Alternatively, as discussed above, conventional techniques may assign an arbitrary label that is not directly related to the audio characteristics of that particular content. The label is inadequate since it may be changed, and is not indicative of the actual content, and thus for cataloging purposes is an ineffective technique for reliably identifying and cataloging a particular content.

Identifying a particular content based on a digital version of that content, can be difficult, particularly when the digital version of the content has been compressed and subjected to communication channel disturbances. Furthermore, since an encoded version of the content usually bears no easily-discernable relationship to the analog version, comparing encoded versions of the content is very difficult. This is usually the result of different encoders employing different encoding algorithms that have differing parameters, such as sample rate, for instance. Accordingly, a preferred embodiment of the invention develops an "analog signature" of a particular content, derived from an analog rendition of the content itself. The analog rendition of an audio stream is similar in content to the audio stream itself.

The present analog signature method, system and computer program product provide an efficient mechanism for coping with the massive size of the decoded stream of audio. An advantage of this approach is that it enables an efficient way to compare two audio streams to recognize when the two streams are in fact slight variations of the same content. This method makes use of analog data, but avoids the computationally expensive task of comparing large amounts of analog data. As a consequence, comparisons between songs need not be performed on the audio data itself, but rather on the "analog signatures" which may be viewed as a small list of properties that reflect the nature of the streams of the audio data.

Figure 4 is a flowchart of a process employed in the present embodiment where the analog signatures are identified and used by the system server (figure 1) to identify whether a particular content has previously been cataloged, and associated with particular attributes. Of course, the "analog signatures", are not limited to the present streaming audio embodiment, but may be used in other applications where large data files are best cataloged based on their content, rather than an arbitrary label assigned to that content.

The process in Figure 4 begins in step 1, where a content to be analyzed is first identified. The content, if in an encoded format, is first decoded, and then a sampled version of the analog format is taken. The analog signature for that content is then determined, as will be discussed in greater detail with respect to Figure 5. Once the analog signature has been produced, the process then proceeds to step S3, where the analog signature is sent from a particular client to the server system. This communication may be by way of the Internet in a digitally formatted message as we discussed with respect to Figure 7. Other communication channels may be used as well, such as wireless terrestrial and space-based

communication channels.

After step S3, the process proceeds to step S5, where an inquiry is made regarding whether an analog signature is in fact present in the server system's data structure, which may be viewed as a special form of a database. If the response to the inquiry in step S5 is negative the process ends, after concluding that the analog signature was not sufficiently close to any other signature that resided in the server system data structure. However, if the response to the inquiry in step S5 is affirmative, the process proceeds to step S7, where content attributes associated with a particular content are associated with the signature and the content itself.

The attributes may include a variety of information categories used for cataloging purposes, such as the artist's name, genre of the music, album name, or play time of the content. Once the particular analog signature has been identified, the server system may then report in the form of a message to the client, the attributes associated with that particular content. Subsequently, the process ends.

Turning to Figure 8, Figure 8 illustrates an audio frequency range that has been subdivided into six frequency bands, as one exemplary embodiment. Other frequency bands may also be employed, such as 16 frequency bands, in a preferred embodiment. The N frequency bands are limited at the high end by the desired amount of processing needed to perform the fast Fourier transfer (FFT) for converting the time domain audio signal into a frequency domain reference. The high end is also limited by the frequency range of the energy in the waveform itself; audio signals can be completely captured below 30kHz, for example. Experimentally, the inventor identified that little additional information may be gleaned by using more frequency bins than 16, for audio information. Of course, additional bins may be preferable where a higher resolution is needed, such as when a greater amount of information is produced in a smaller bandwidth. The number of frequency bands on the low end (i.e, a small number of frequency bands), limits the signature's resolution and thus reduces the number of dimensions used to uniquely characterize the composition of particular content. As a consequence, the space between different analog signatures, when saved in a data structure, will be reduced if the number of dimensions are reduced.

By breaking the audio band into N frequency bins, each frequency bin may be handled independently so that an intensity of the audio energy in each bin may be characterized over the entire song duration. The information of the content may not be, and

probably will not be, independent between the bins, but the values representing the signature for each band may nonetheless be handled as if they were independent.

A song to be characterized is first passed through a CODEC so as to translate the digital information into an analog waveform. Thus, once the audio energy is produced in a format suitable for playing through a computer-speaker for example, the sampled data stream is applied to a FFT mechanism so as to transform the time domain sequence into a frequency domain representation. The output of the FFT, takes a predetermined number of sample points at a time in order to produce corresponding number of other points. In the preferred embodiment, a 2048 point FFT is used, with a lesser amount of points losing some fidelity in the high end of the content. The result of each FFT operation is viewed as a set of points, that fall within each of the different bands. In this way, it is now possible to deal with each frequency band independent of one another so that the different FFT points that fall within each band may be combined in various statistical ways.

In the present embodiment the points that fall within each band over an entire song are averaged and a standard deviation of the points is determined. As a consequence, for each band, there are two values used as part of the signature, the first value being an average intensity (S_i) and a second value being a standard deviation (S_d) of the average spectral points in each band. An analog signature is denoted as $S_i(1, x)$, $S_d(1, x)$, $S_i(2, x)$, $S_d(2, x)$, . . . $S_i(n, x)$, $S_d(n, x)$, where there are "n" bands and "x" represents the content. As a consequence in the preferred embodiment, there are 32 numbers used as an analog signature to identify a particular content. This analog signature may then be saved in memory for subsequent retrieval, processing or distribution.

The flowchart in Figure 5 describes a process by which an analog signature may be developed. The process begins in step S11 where a particular content is identified in an encoded format (an optional step) for being subjected to an analog signature process. The process then proceeds to step S13 where the content is decoded through a CODEC so as to obtain a time domain signal. The process then proceeds to step S15, where a FFT is performed on the time domain signal so as to develop X spectral points per FFT operation. The number of FFT points is a function of the processing availability for handling the FFT operation. Other transforms may be used as well such as discrete cosign transform or other recognized approaches for characterizing a time domain signal in the frequency domain.

After step S15, the process proceeds to step S17, where the X spectral points produced in the FFT operation are distributed into the N frequency bands, where in the preferred embodiment N is equal to 16 frequency bands, each having a width that progressively increases by a power of 2. Other widths for the frequency bands may be used as well, such as uniformly spaced widths, exponential incrementation and the like.

After step S17, the process proceeds to step S19 where an average is calculated of all the spectral points falling within a particular band so as to identify for each band, an average intensity for the content in that band (S_i). After step S19 the process proceeds to step S21, where the statistical measurement of the variation in sample distribution within a band is calculated. In the present embodiment the statistical measurement is a standard deviation based on the sample points for each of the frequency bands (S_d). The process then proceeds to step S23, where the digital signature S_i (n, x), S_d(n, x), is saved in memory. The process then ends.

By having an analog signature of a particular content, it is then possible to quickly compare one song to another without having to look through many megabytes of data that represent the actual waveform itself. Furthermore, because each song (which is the content in the preferred embodiment) is represented by a 32 number representation (analog signature), it is now possible to compare two songs in a quick fashion, where even if there are differences between the songs, the Euclidean distance between the signatures will not be very great if the songs are similar. Thus, distance may be used as a metric for comparing similarities between different songs so as to identify whether two songs are the same, or even have similar pattern associated with them. The present inventors recognize that this distance measurement, may be used to detect the presence or absence of particular songs, as is done with digital communication devices when determining whether a particular channel symbol has been received at a receiver (which uses distance measures as a way to determine a signal to noise ratio).

The distance measure, a Euclidian distance, has four properties that are met by the current process. A distance from any point A to itself is always 0. The distance from A to point B is not negative. The distance from A to B is symmetric with the distance from point B to point A. And the last property being that the triangle inequality property is met.

When comparing one piece of content to another, the distance measure could be

performed by determining whether two signatures are similar by comparing the differences of the averages and the standard deviations for each frequency band. However, for the distance metric, it is desirable to find an appropriate way to combine the differences. This can be done by treating each value as an independent quantity (even though they are not strictly independent), putting the quantity in a space of 2N dimensions (where each dimension is defined by either an average or a standard deviation of a frequency band), and finding the Euclidian distance between two points (i.e., the square root of the sum of the squares of the coordinates in the 2N-space). Accordingly, the distance between two points may be represented by

$$D(x,y) = \sqrt{\sum_{n=1}^{N} \left(Si(n,x) - Si(n,y)\right)^2 + C\sum_{n=1}^{N} \left(S_d(n,x) - S_d(n,y)\right)^2}$$

where C is a constant use to scale an effect of the standard deviations on the function. Moreover, in order to compensate for slight regular differences in data that result from different CODECs, several transformations are made on the signatures to normalize them in a way that the distance between two signatures is minimized.

A user that has a collection of music on a computer may wish to build a profile. In this case, it is preferable to have accurate information regarding the actual music listened to by the user so a program may go through that user's collection of music, create a signature for each song, and send that set of signatures to the server system so that the client need only send a very small amount of information to the server in order for the server system to look up each signature in its database and identify whether a particular signature corresponds to a song or has already been associated with attributes about that song that are saved in the server system. Information and attributes regarding different songs (content) is generally a manually intensive process that involves accurately labeling collections according to artist, song name, album the song is on, genre, and whatever other information that may be of interest. That information may then be sent back from the server system to the client so that the user of the client automatically receives this information for use as that particular person sees fit. Having these attributes associated with the songs listened to by the user, will help

the system develop a dynamic profile regarding that particular user's listening interest and patterns.

Figure 7 is a data structure showing the functional attributes of a digital message that includes an analog signature developed by a client and sent to a server system. Of course, the server system may also distribute the analog signature to other systems, including the client, and is not limited to a one-way distribution from the client to the server system. The data structure includes a header 701 that identifies the different parties in the communication transaction as well as separate subfields, where each subfield holds either a signal intensity, or a signal standard deviation for a frequency band used to characterize the content. Subfield 703, for example, includes a signal intensity (preferably in dB) for a first frequency band of a content labeled "X". Subfield 705 includes a standard deviation for the first frequency band for the content X. Likewise, subfield 707 and 711, as well as alternating subfields therebetween include signal intensities for the second frequency band through the n^{th} frequency band, respectively. Similarly, subfield 709 includes the standard deviation for the second frequency band and subfield 713 includes the standard deviation for the sample points that fell into the n^{th} frequency band. As was the case with the signal intensity sub-bands, there are intermediate subfields between subfields 709 and 713 that include the standard deviations measured in the frequency bands between the second and the n^{th} frequency bands.

HEADER LOCKING

Building a profile for a user requires knowledge of what content (e.g., songs, video, etc.) the user actually plays. That knowledge includes a number of attributes associated with each piece of content. Since profiles are preferably built at the client, and since it cannot be presumed that users will accurately label the content, a mechanism for labeling the content by the server is preferred. Due to the nature of digital media, it is possible for content to be copied multiple times without any loss of fidelity. Since one piece of content may be proliferated widely, if a set of attributes were attached to the original piece of content so the knowledge is proliferated along with the content itself, the content-labeling servers would not receive future queries concerning that piece of content.

As presently recognized, a problem may arise from simply attaching extra information to that piece of content. Since there is not necessarily any inherent bond between the content

and the attached attributes, it would be possible for a malicious party to attach a false set of attributes to a piece of content. If that content were then proliferated, the profile of any user who plays that content could be adversely affected and thus not reflective of that particular user's taste.

Using a header locking technique according to the present invention, it is possible to create a bond between the attached information and the content itself so as to make it extremely improbable that an unauthorized party could attach information to a piece of content and have that information not be detected as having been tampered with. In particular, a piece of content C may have a set of attributes A, and a transformation operation T_s, while T_{c1} and T_{c2} are transforms performed by the client. T_s, or some element thereof, is a secret known only to an authorized content labeling authority. When C is labeled, A is attached to C at that time. T_s may be performed on A and C to get some result T_s (A, C). That result may be attached to C as well. Once these operations have been formed, the information may be distributed without any concern of data integrity as [C, A, T_s (A, C)].

When a client receives [C, A, T_s (A, C)], the client extracts C and A, and computes T_{c1} (A, C). The client then extracts T_s (A, C), and computes T_{c2} (T_s (A, C)). If T_{c1} (A, C) matches T_{c2} (T_s (A, C)), the client may then trust that A is from an authorized labeling authority. T_s (A, C) may be named as the "header lock", since A will often be placed in a header of C and T_s (A, C) serves the purpose of "locking" A to C.

The strength of the header lock depends on the properties of T_s, T_{c1} and T_{c2}. In a preferred embodiment, T_s corresponds to hashing a document and then encrypting the hash with a private key while T_{c1} corresponds to hashing a document, and T_{c2} corresponds to decrypting a file with a public key.

The header lock process according to the present invention differs from a conventional digital signature system in that the present system binds two objects that essentially have no inherent bond. More particular features of header locking system, method and computer program product are described below.

Before discussing the specific features of the header locking embodiment according to the present invention, a brief review of conventional techniques for using digital signature for protecting the integrity of data files is in order. The conventional process is shown in Figure 9, the first step in which, step S51, is to calculate a hash at a source terminal from a file that is

to be signed with the digital signature. Different types of hashes include MD5 and SHA for example. Conventional hashing operations are described in Schneier, B. "Applied Cryptography: Protocols, Algorithms, and Source Code in C", 2nd Ed., John Wiley and Sons, 1995, ISBN: 0471117099, the contents of which being incorporated herein by reference. Accordingly, the file is processed by the hash function in order to come up with a hash that would be a small representation of that particular file. The process then proceeds to step S53 where a digital signature is performed by encrypting the hash with a private key (K_{pri}). The RSA (Rivest, Shamir, and Aldeman) public key cryptosystem is an exemplary digital signature that may be employed, or other ciphers may be used as well, as discussed in Coutinho, S. C., "The Mathematics of Ciphers: Number Theory and RSA Cryptography", A. K. Peters, Ltd., 1998, ISBN: 1568810822, the contents of which being incorporated herein by reference.

The digital signature is then sent along with the file to the destination terminal in step S55. The process then proceeds to step S57, where the signature is decrypted at the destination terminal using the public key of the client so as to obtain a "hypothesis hash". The destination terminal then calculates the hash from the file itself in step S59 and determines whether the hash is equal to the hypothesis hash in step S61. If the response to the inquiry in step S61 is affirmative, it is clear that the document has not been tampered with, or otherwise corrupted so the file may be processed according to the end application in step S63. However, if the response to the inquiry in step S61 is negative, the process proceeds to step S65, where it is recognized that the file has been corrupted, by some mechanism and as a consequence, a process is initiated for handling the corrupted file. This may include sending a reply message to the source terminal indicating the source terminal needs to send another copy of the file. Then the process ends.

The process, system and computer program product of the present embodiment discussed below, is able to create a bond between two objects that have no inherent bond, and therefore add another level of integrity to headers that are appended to content so that attributes associated with that content may be protected for mass distribution over, perhaps, the Internet. The process in Figure 10 begins in step S71, where at the source terminal separate hashes are calculated for a message header, HASH (header) and for the content HASH (content). The process then proceeds to step S73, where the HASH (header) and

HASH (content) are combined in some fashion, such as a concatenation process, boolean logic process, or other process where the content from both HASHs are combined in a fashion where the respective HASHs may be recreated at the destination terminal. Once combined, the process proceeds to step S75, where the combined HASH is encrypted with a private key (K_{pri}) so as to create a header lock at the source terminal.

In step S77, the header, header lock and content are sent to a destination terminal by way of a locked message. A data structure for such locked message which may be saved in memory before being transmitted will include a header, header lock, and content, in separate subfields. At the destination terminal, in step S79, the destination terminal calculates hypothesis HASHs for both the HASH (header) and HASH (content). The process then proceeds to step S81, where the destination terminal forms another combined HASH by combining the HASH (header) with the HASH (content) in the same fashion as was done at the source terminal. The process then proceeds to step S82 where the destination terminal decrypts the header lock that was included in the message from the source terminal to the destination terminal by using the public key (K_{pub}) so as to obtain the hypothesis combined HASH. The process then proceeds to step S83 where an inquiry is made regarding whether the combined HASH (formed in step S81) is equal to the hypothesis combined HASH (step S82). If the response to the inquiry in step S83 is affirmative, the process proceeds to step S84 where the file is processed according to normal procedures, because it is concluded that the integrity of the file is intact. However, if the response to the inquiry in step S83 is negative, the process proceeds to step S85 where another process is performed under which it is presumed that the file has been corrupted. In this instance, the destination terminal sends a reply message back to the source terminal informing the search terminal that the file was corrupted so that the source terminal can issue another message. Subsequently the process ends.

DYNAMIC PROFILE

Profiles are often developed to characterize consumers' buying trends and tastes. The concept is to develop a description about the trends of a particular consumer's interests in consumable goods and associate that information with personal information about that user. Sometimes, only the personal information is collected about the consumer. This profile

information may then be sold to advertisers so the advertisers may communicate directly with the consumer or resell the information to other advertisers who seek to profit from the information. Development of buying trend profiles presumes that the customer's tastes exist at all periods of time, and thus is not applicable when there exists a stream of transactions where the results of one transaction can affect the next transaction.

The present inventor recognized that adding a "dynamic" aspect to a profile of a user's trends would be beneficial in various applications, such as the present embodiment where streaming audio and video is received by a user, where the perceived musical or video tastes of a user may vary according to certain factors, such as time, previous events, or sensory input (e.g., heart beat). In the example of the present embodiment where the user employs an audio player at the client, the dynamic profile may help to suggest a next song in a play list may be for a particular user having musical tastes that are reflected in the dynamic profile. For example, the user may not opt to listen to classical music very much, but when that user does in fact listen to classical music, that user typically prefers to listen to several classical musical works in a row. Dynamic profiling would thus take the user's recent history into account (by retrospectively observing the user's listening pattern) in such a way such that if the user had in fact recently listened to several classical songs, the dynamic profile would then point to another classical song as an appropriate recommendation for the next song in the play list. If, on the other hand, the user enjoyed classical music, but never listened to two classical works in a row, the dynamic profile would not recommend a classical song if the user had just listened to a classical song.

While the present embodiment is directed to observing and basing association rules on a user's pattern of having completely listened to different musical works, the dynamic profiling aspect of the present invention, is applicable in a number of other applications as well. For instance, the user's listening pattern may be influenced by a number of other discrete, environmental influences, such as the time of day, season, weather, location of the user (e.g., in the car or at home or office) and biosensory input, such as heart beat, eye movement and the like. For example, if the biosensor detects little eye movement or the eyes having been closed for a certain period of time, the next song in the play list may be a classical work. Furthermore, if a light sensor determines that the amount of light has increased at a predetermined rate, and the eyes have not opened for a certain period of time,

then the next song in the play list may be a song that the user enjoys when waking up in the morning.

The flowchart of Figure 11 describes a process employed in the audio player when using a user-actuated dynamic profile operation. Moreover, in the player, the user, under user control may opt to select a dynamic profile to be created to suggest additional songs to be played in a play list. The songs may be either hosted locally on the client, or retrieved from a database that is connected to the server system. Alternatively, the client may connect to other clients or other digital processing equipment, as was discussed previously with regard to Figure 1 to obtain the audio files. Through a graphical user interface, the user begins the process in step S101, where the user indicates whether the user would like to select a particular profile. These profiles (one or more), are saved at the client, and in alternative embodiments also saved on the server system. The user may edit or delete the profile as the user sees fit, and may even merge the user's profile with other profiles from other people, or other profiles relating to that user, that were created in the past.

The process begins in step S101 where if the response to the inquiry in step S101 is negative, the process proceeds to step S107, where the user manually selects a next file (e.g., an MP3 file) in order to populate a play list. Subsequently the process proceeds to step S110, where that file is processed, which in the embodiment of an audio player, would result in the audio file being played through a speaker. Subsequently the process ends.

However, if the response to the inquiry in step S101 is affirmative, the process proceeds to step S103 where an inquiry is made regarding whether a song can be selected from a dynamic profile (based on the song being recognized as having a sufficiently high probability). If the response to the inquiry in step S103 is negative, the system attempts to implement a static profile, in step S109. If the song may be selected with the static profile the song is selected in a linear fashion because the static does not include a retrospective analysis of the user's past listening trends based on a limited number of immediate past listening events. If the song cannot be selected from the static profile, the song is selected randomly from the playlist. Subsequently, the process proceeds to step S110 where that file is processed. However, if the response to the inquiry in step S103 is affirmative, the process proceeds to step S105, where a dynamic profile is created, as will be discussed in more detail below, from a subset of past discrete events that relate to a user's listening trends. After the

dynamic profile is created, and association rules are developed, then a next song is selected based on that dynamic profile having indicated that the next song would be the most likely candidate giving the pattern of past events. After step S105 the selected file is processed in step S110 and then the process ends. Alternatively, the process may be repeated where the dynamic profile process, or static profile or even the manual selection of files is repeated until the user decides when to discontinue the automatic or manual song selection process.

In conventional players, the user is provided with a play list. The user can either select a pattern of play, such as a sequential playing of songs on the play list, or perhaps a random selection of songs on the play list. By implementing profiling, either static or dynamic, the user need not randomly, or sequentially select files, but rather may rely upon the player's perceived assessment of the user's unique preferences (in an absolute sense with regard to the static profile, and in a "context-sensitive" manner with respect to the dynamic profile). The static profile is based on a probability analysis of the user's overall perceived listening habits, in relationship to attributes of songs included on a play list, or in a larger data base of content available for download, perhaps from the server system. The dynamic profile is more comprehensive, discussed below, and is based on an association of subsets of past song sequences listened to by that user.

The dynamic profile mechanism and process are based on an understanding of association of rules. The discussion below will relate to association rules to some extent, however a general tutorial on the subject is not believed to be in order, since there are a number of references that discuss association rules generally. For example, Mannila, H. et al, "Discovering Frequent Episodes in Sequences" (see, e.g., Appendix C) is a paper that discusses profiles that have a temporal aspect. A ``frequent episode" is a partial ordering of events that occur in a data stream. Because full partial orderings must be constructed from the data, building frequent episodes is computationally very expensive. Since the dynamic profile component of the present invention includes only an antecedent and a consequent, the dynamic profile process according to the present invention is not as computationally intensive as frequent episodes. This article, as well as the four articles discussed below and included in Appendix C are part of the present specification. Another article on the subject, is Lane, T., et al, "Sequence Matching and Learning in Anomaly Detection For Computer Security". The type of sequencing described in this article is directed to the notion of allowing a system

determine whether a sequence of events is likely or not. However, due to the way the sequences are constructed, there is no good way to use this information to predict what should come next based on a recent history. These sequences can only be used to say whether a sequence was normal *a posteriori*. A third article is Agrawal, R., "Mining Sequential Patterns" explains how to find sequences of events that are common to several independent data streams. The dynamic profiling process according to the present invention finds sequences of events that occur within a single long data stream. Further, the sequences found by the technique described in the article potentially occur over a large span of time, and thus are not as appropriate for predicting an immediate next event from a recent history. A next article is Ramakrishnan, S., et al, "Mining Generalized Association Rules". In this paper, association rules are discussed. Association rules have do not have any mechanism for dealing with the temporal component of data. The last article is Lee, W. et al, "Data Mining Approaches For Intrusion Detection". This article, and in particular section 3.1, provides a brief overview of association rules. Section 3.2 describes frequent episodes in the context of providing time-sensitive profiles. The first article discussed above, "Discovering Frequent Episodes and Sequence", also is directed to frequent episodes.

Dynamic profiling in the present embodiment, is different than conventional dynamic profiling, in that with conventional techniques, frequent episodes are constructed for the purpose of discovering partial orderings over sequences of data. Episodes are things within the frequent episodes that might happen in parallel or in serial. Thus, in conventional dynamic profile applications, there are sequences from which a directed graph may be built in order to support this partial ordering operation. In the context of the present invention, however, a more efficient process is developed in that the data to be analyzed occurs in the past (an antecedent), and observes what comes after the inquiry (the consequent). This efficient process, it is better suited for situations in which the dynamic profile is employed to predict the next event based on prior events. Furthermore, the present embodiment of the invention, avoids the need to build an entire partial ordering out of this data, and therefore is much more aptly suited for clients that have limiting processing resources.

A process for implementing a dynamic profile is discussed with respect to Figure 12. More detailed substeps that occur in selected of the steps will be discussed after Figure 12 is introduced. The process begins in step S91, where a user history of discrete events A, B, C,

..., is recorded in memory. Discrete events for the present embodiment may be such things as the completion of listening of a particular song. However, in other embodiments where the dynamic profile is used to support the actions of other agents, the discrete events may be manual activity, such as getting in a car, sensory input or the like. For example, the dynamic profile may be used in this instance to automatically initiate a telephone call to an insurance company, when sensors indicate that a residence has been exposed to a hail store, or other damaging event. Also, dynamic profiling may be appropriate for initiating communication with other entities, such as food delivery ordering, alarm conditions and the like based on sensors that indicate a present state of being of an activity that's being monitored.

Labels assigned to the discrete events (indicators) are saved in memory at the client, so that an array of transaction data may be kept. In the present embodiment the array of transaction data is stored in a time-based fashion, where events that occur sequentially in time, are stored in a predetermined logical relationship in the array, such as in adjacent addresses. The process then proceeds to step S93, where a window of size N+1 is passed over the transaction data (which in the present case are songs that have been listened to by a user for which the profile is being created). N is the "context size", and thus captures the discrete events that happened prior to the "last element" that occurs in the window. For example, a window of size 3 (2+1) may pass over discrete events ABC, where "A" and "B" are the "context" for the result "C". In this situation, A and B are described as being the "antecedent", and C being the "consequent".

The process then proceeds to step S95 where the "last element" for each transaction is marked as being a last element, i.e., the consequent. It should be noted that steps S93 and S95 may be repeated for differing size windows that increase progressively in size so that the association rules may be based on not only two discrete events as identifying the context, but also 1, 3 or even more events defining the context.

After all the transactions have been identified by making sequential passes, of varying window sizes, over the transaction data, the process proceeds to step S97. In step S97, every one of the discrete events is expanded according to the attributes associated with that event, based on a predetermined taxonomy. This taxonomy describes the attributes of every event and attributes of those attributes etc. For example, in the context of the preferred embodiment, the taxonomy may be that for a certain song X there may be a certain artist, as

well as a certain album on which the song appears, and the song also having a certain genre. Thus, by playing the song X, would correspond with an event A, and event A may be expanded into A1, A2, and A3, in association with its three attributes. This expansion is performed for each of the discrete elements in the window as shown in step S97 of Figure 12. The last elements in the windows, even after being expanded, are signified (i.e., represented in lower case in the example of Figure 12) as being the last element . The reason why this last element is identified, is so that association rules may be developed where prior events on positioned on the antecedent side of the rule (e.g., left side), and the last element is on the consequent side of the rule (e.g., right side). Since some attributes may be redundant, redundant elements are omitted, so as to not upwardly bias the significance of one particular pattern of events.

After Step S97, the process proceeds to step S98, where a set of association rules are developed by treating each of the windows as a transaction. Details of step S98 will be discussed in more detail below. After the association rules have been developed, the process proceeds to step S99, where rules that contain last elements as an antecedent, are discarded, to the extent they still exist, if the redundant elements were not omitted in step S97. Furthermore, where the consequent contains elements that are not the "last element", these rules are discarded as well. After step S99, a complete set of association rules are developed and so the process ends.

Examples of association rules developed using the above-identified processing mechanism will now be described. A format of the rules is as follows:

[confidence support] Category 1 = A Category 2 = B Category 3 = C and the consequent (-->) is attribute D.

A particular example of an association rule developed may look like the following:

[0.6153 0.0516] genre = pop genre = rock album = the fury of the aquabats! --> genre = ska.

This association rule would indicate that 5.16% of the time, the user listened to a set of songs that contained the attributes genre = pop, genre = rock, and the album = the fury of the aquabats!. Then, as a consequential action, the user listened to something with the attribute genre = ska. Furthermore, this association rule indicates that 61.53% of the time, that the user listened to a set of songs that contained the attributes genre = pop, genre = rock, and album = the fury of the aquabats!, the next song the user listened to had the attribute genre = ska.

Another example of an association rule may look like

[0.5555 0.0322] group = Sara McLachlan group = Tori Amos = = < = genre = pop.

This association rule would indicate that 3.22% of the time that the user listened to music by Sara McLachlan and Tori Amos (in any order) then the user preferred to listen to something from the pop genre. Furthermore, 55.55% of the time that the user listened to Sara McLachlan and Tori Amos, the user actually listened to something from the pop genre. Based on these association rules, the confidence and support may be compared against predetermined thresholds so as to indicate whether particular songs having particular attributes would be of sufficiently great interest to the user so as to include those next songs in the user's play list.

The present inventor recognized that a limitation with a conventional approach to determining whether a particular candidate item set of a particular size in fact occurred in the transaction data would be extremely time consuming since the size of candidates could be quite extensive, given that the number of attributes, and the size of the candidate item set may grow large. For example, there may be 60,000 or more candidate item sets (or attribute sets). Thus, comparing each item set against the candidate item sets would be extremely time consuming and computationally intensive. In light of this recognition, the inventor identified the process described in Figures 13 and 14 as to be a much more efficient way to develop association rules from the item sets, based on this type of transaction data. The process shown in Figure 13, corresponds with the process steps shown in Figure 12, and in particular, steps S93-S98.

Referring to Figure 13, a process step of S112 relates to making a first pass over the transaction data with a window of size Z (which in the first case may just simply be a window of size 2, having one antecedent discrete element and a last element). By performing this step, individual attribute sets (or item sets) that occur over a probability of X (a predetermined probability) may be identified. A first subset, step S222 creates a HASH table, such as HASH table 803 in Figure 14. This HASH table contains pointers associated with certain attributes, where the pointers point to a list of candidate item sets 805 in which that particular attribute is found. For example, referring to Figure 14, if a window of size 2 (802) includes an attribute A, then the different pointers that are included in the HASH table 803, point to the different candidate item sets in table 805, that include the attribute A. Another

table (not shown), is then kept so as to keep a count of each of the candidate item sets identified in table 805. The count is a cumulative count while the window 802 passes over the entire transaction data 801. Thus, the process of implementing the separate counters for each occurrence of the candidate item sets is performed in step S113 (Figure 13).

After step S113, redundant attributes are deleted in step S114, so as to avoid a false count. For example, suppose attribute A1 was actually the same attribute as B2. This may occur if the user listens to two songs by the same artist, they would have the same attribute. In this case, that artist is only listed once. The process then proceeds to step S115, where a histogram (or cumulative count for each item set) is developed for the occurrences of the different attribute sets for that particular window size. Subsequently, the process proceeds to step S116, where another pass is made over the transaction table 801, with a different size window so that a different size set of attribute sets may be identified that occur greater than a certain predetermined probability. The process then proceeds to step S117, where further passes over the transaction data may be performed.

Based on the histograms that were developed on the occurrences of the different sizes of item sets, association rules are then developed from the item sets and the support for each of the item sets is identified (probability of occurrence) as well as the confidence (likelihood of consequent occurring given the antecedent occurred). Subsequently, the process proceeds to step S119, where the probabilities are normalized so as to compensate for redundancy created by the expansion of events into attributes. Features of this later step will be discussed in greater detail with regard to Figures 15A-15C and Figure 16. Then, the process ends after step S119.

One of the benefits associated with the process implemented as described in Figures 13 and 14, is that using the HASH table, allows for the searching to be done in constant time. Searching through the candidate list, however, is done in linear time. Accordingly, by using the HASH table, the number of items that need to be searched in the candidate list, is greatly reduced, thereby allowing for a minimal amount of computational processing, and time needed to create the dynamic profile and the association rules that support the dynamic profile.

ITEM SET CHOOSER FOR SUPPORT SETS OVER HIERARCHICAL DATA

In developing the dynamic profile, the different discrete events were expanded according to certain attributes. Static profiles on the other hand are merely a support set although each transaction consists of a single event that includes all the attributes associated with it. The dynamic profile however includes the number of attributes, some of which may overlap. Due to the taxonomical nature of the attributes associated with an item, the expansion of a discrete event into its respective attributes, may result in a false indication that support for the item set is actually greater than it is. This point may be illustrated best with regard to the venn diagram 830 in Figure 16.

In Figure 16 a first artist, Artist No. 1, is represented by space 831, which contains Album 1823 and Album 2835, both of which were prepared by Artist No. 1. Song No. 1833 was produced by Artist No. 1 and occurred in Album No. 2835, as well as in space outside of Album 2. Furthermore, a second artist, Artist 837, may have performed with Artist No. 831 in some overlapping space. In particular, some songs 838 prepared by Artist No. 2 may have also been prepared and performed by Artist No. 831. Accordingly, in considering these different attributes shown in Figure 16, the attributes are overlapping in nature. Accordingly, due to this overlapping nature all the item sets in the support set, will have supports that are artificially high. Another way of stating this is that if all the supports are added up, the supports will result in a value that is greater than 1, which is not the case since some of the areas overlap one another.

The above discussion will now be presented from a different perspective, referring to the data as being structured in a hierarchical nature. Certain types of data mining processes require looking at data as a series of transactions. An item set is a set of items that could possibly appear in a transaction. The support set of the data is the set of item sets that occur in at least a minimum number of transactions. An item set that has K items is called a K-item set.

Because the discrete events have been expanded according to their attributes, the expansion is done according to the taxonomy of the attributes. This taxonomy is hierarchical in nature. A grocery example will be provided to help illustrate this point. Whole milk might be an item that appears in a transaction. In that case, whole milk is a leaf-node (terminal node, i.e., a node that defines a most specific attribute) in a hierarchy where milk is a parent of whole milk (as well as skim milk, 1% milk and 2% milk), and dairy products is a parent of

milk (as well as cheese and butter). Thus, any time a customer buys whole milk, that transaction will expand to contain whole milk and milk and dairy product. Building support sets and incorporating hierarchies with support sets are what is focused on in conventional profile systems.

However, when a support set is created with hierarchical data, using that support set to suggest items for future transactions to a user can be problematic. For example, a supermarket may have built a support set over many transactions for a particular customer. The supermarket may find that this customer purchases whole milk in 20% of the purchaser's transactions, buys milk in 25% of its transactions, and buys dairy products in 50% of his transactions. If the supermarket used this data to make special offers to the customer, the supermarket may make special offers on dairy products 50% of the time, it may make special offers on milk an additional 25% of the time and it may make special offers on whole milk an additional 20% of the time. Since whole milk is a dairy product and since a special offer on dairy products would presumably apply to whole milk, the customer will receive special offers on whole milk and dairy products more frequently than the customer had traditionally purchased those items. The assumed independence of hierarchical related data can produce results that do not reflect the nature of the data used to build the support set. One of the deficiencies with this approach is that while building the support set, independence is assumed between all items.

Accordingly, in the present system and method according to the present invention, dependence is purposefully introduced by building a series of directed graphs out of all the items. Each directed graph has a property that traveling from one node to the next results in an increase in specificity. Furthermore, if there is some item set A that is more specific than some item set B, and if some item i is an element of A, then i is also an element of B. Thus, there is a route from B to A. In the above example, since the hierarchical taxonomy is known for the item sets, dependence can be introduced after the fact.

Using the grocery example, milk may be viewed as being more specific than dairy products, so whole milk is more specific than milk. Thus, the graph is directed from dairy product to milk to whole milk (i.e., less specific to more specific). Now, it is possible to determine the independent supports of each item set; the support of an item set is the frequency with which it appears in the transaction data while the independent support is the

support of an item set independent of other item sets. Accordingly, the independent support of whole milk is 20% (the same as the support, since whole milk is a terminal node of the directed graph. The independent support of milk is 5%, and thus it must be recognized that milk excludes whole milk. Moreover, the independent support of milk is derived from subtracting its support from the independent support of the item set that is more specific, and explicitly noting that each item in more specific item sets are no longer included in the milk item set. Using this approach, the same process may be applied to other dairy products' item sets so as to determine the independent support of the dairy products being 25%. After the directed graph is completed, the supermarket may then make special offers based on the graph and the offers will reflect the actual buying habits that produced the original support set. While the present description describes actually making a graphical representation, the term "directed graph" as used herein need not be a pictorial representation but other representations that perform a similar function.

Thus, a feature of the present method is to build a directed graph with nodes corresponding to item sets with greater degrees of specificity resulting from following the edges of the graph, or its equivalent. Another feature of the inventive process is that a unique technique is used to form the directed graph. In the supermarket example above, all item sets were one-item sets and the increase in specificity came by following the predefined hierarchy. In actual systems, however, finding item sets with greater specificity can be much more complicated. For some K-item set T, to add T to the graph, the nodes that are more specific need to be looked at in any one of three ways. These three ways are described in Figures 15A, 15B, and 15C.

Referring to Figures 15A, 15B, and 15C, in each case, a directed graph is constructed out of the support set and taxonomy such that three types of connections exist between nodes. These directed graphs do not include cycles. Also a node that has no connections leaving from it is referred to as a terminal node. The independent support of a terminal node is the same as its support. The independent support of a non-terminal node is equal to its support minus the sum of independent supports of all nodes accessible by following connections from that node. Accordingly, one type of directed graph that goes from less specific to more specific is shown in Figure 15A. In this case, each of the item sets 811, 812, and 813 and 814, have all the elements of the parent item set 810. Another type of directed graph that

goes from less specific to more specific is shown in Figure 15B. The nomenclature employed in Figure 15B, uses apostrophes to indicate that an attribute with an apostrophe is a taxonomical direct child from the attribute that does not have the apostrophe. Thus, for the item set 817, child item sets 815 and 816 are more specific since item set 815 includes a taxonomical direct child from an element in 817 (A' is a direct child of A), and item set 816, is more specific than item set 817 (because attribute B' is a taxonomical direct child of B from item set 817).

Figure 15C shows a third type of connection in the directed graph in which item sets having fewer elements may be more specific than the parent item set. For example, item set 821 or 819 may each be more specific than item set 823. This would occur when both A and B, the attributes in item set 823, are both parents somewhere in the taxonomy of either X or Y. Identifying the independent supports for each item set, helps to avoid the problem of a convergence of the profile, thus contaminating the profile of the user. For example, if the normalization was not performed so as to identify the independent supports, then the profile would be populated with selections in a repeating pattern so as to contaminate the statistical probabilities of the events that form the basis of the user's actual profile. As a consequence, repeating the selection of a same song would tend to move the profile in a certain direction until the profile gets into a certain state where the user's preferences would tend to converge on one or two pieces of content. Employing the normalization process through the item set chooser for support sets over hierarchical data avoids the problem of convergence and thus does not self-contaminate the profile when the dynamic profile is in fact being employed.

MEDIA AGENT PRIVACY ASSURANCE

One feature of the present invention is that it provides a method and a mechanism by which a user of a client is able to maintain anonymity throughout the use of the system. Furthermore, the user remains in control of any user-specific information that the user may consider confidential. This is unlike conventional systems where the user loses control of the user's private information when the information is transferred to a server system. In contrast, the present invention provides a mechanism by which the client remains in control of the user's information and controllably releases the confidential information as the user sees fit.

As seen in Figure 17, a client 850, includes a computer-based media agent. Figure 18

shows the same thing, but explains that the client 850a may be a mobile agent (e.g., a cellular telephone, pager, personal digital assistant or a display in an automobile, for example). This media agent may be a personal computer, or a mobile device that includes a processor, and is capable of communicating with external devices. For example, the media agent may be employed in the electronics of a car, on publicly-available platforms, such as airplanes, trains, ships and the like. The physical location of the media agent is of secondary importance to the function performed by the media agent.

In particular, the media agent provides a mechanism by which the user is able to employ a dynamic profile mechanism 856, as previously discussed, to develop a profile of a particular user based on a series of transactions implemented by that user. In this case, the transaction profile is saved as a component of the dynamic profile mechanism. These transactions may relate to a preferred set of music listened to by the user or other events (such as purchased items) that have been recorded in the transaction table. On the other hand, the dynamic profile mechanism may develop and hold demographic profile information regarding that particular user. In Figure 17, separate profiles (transaction profile and demographic profile) are included as part of the dynamic profile mechanism. Alternatively only one profile may be included.

The user has access to the demographic profile as well as the transaction profile and may edit the same in order to adjust profile features that the user wishes to modify. Although not shown in the figure, the media agent provides a mechanism by which features and attributes within the different transaction profile and demographic profile are presented to the user so the user may edit those profile attributes based on the user's request. However, as will be discussed, when the user actually edits the demographic profile or the transaction profile, the corresponding profiles held by the secured profile data base 874 at the service system 1, will not reconcile with the edited version saved on the media agent. Accordingly, the server system 1, is able to hold unmodified versions of the transaction and demographic profile data that may later be used to determine when manual-edits have been made to those profiles.

As will be discussed, the difference between a manually-altered profile, and an unaltered profile, is that the unaltered profile reflects a series of discrete events, relating to that particular user. On the other hand, in the case of the altered profile the pattern of activity and linkages between antecedent and consequent events, may not reflect the user's actual

preferences and tendencies. As will be discussed, altered profiles may provide third parties with a false indication of that particular user's preferences, if that user decides to share, match, or exchange the profile others.

An input/output device 854, connects to both a dynamic profile mechanism 856, as well as the privacy assurance mechanism 852. The dynamic profile mechanism 856, may pass the transaction profile or a demographic profile through the I/O mechanism 854, although with the optional security added by the privacy assurance mechanism 852. The privacy assurance mechanism 852, holds at the media agent, a private digital signature key that is used to encrypt message traffic sent through the I/O 854, which may include the profile data. Accordingly, the privacy assurance mechanism 852 digitally signs the profile information sent from the dynamic profile mechanism 856, under user control. Accordingly, when the information is sent over the network 5, by way of the client-controlled communication link 858, the service system 1 may then employ a public key to assess whether or not the information had been tampered with. As an alternative embodiment, a profile data is not kept on the service system 1, and all profile data is kept on the media agent. In this situation, the media agent 852, controls both the private and public key, when encrypting the profile data that is sent to the service system 1, for subsequent distribution to other parties. Thus other parties, as well as the server system will rely on the media agent 850 to provide whatever results the server system or other parties wish to glean from the profiles.

One aspect of the privacy offered to the user of the media agent is that the user maintains control over the communication link 858 because the user initiates the communication session with the service system 1. Moreover, the server system 1 does not retain address information for initiating communications with the media agent 850, but rather relies on the media agent to initiate communications with the server system 1. Accordingly, the server system 1 is not configured to send information to the media agent, unless the media agent first initiates communications with the server system 1. This enables the user to avoid the user from being subjected to message traffic and unwanted advertising sent by advertisers.

The server system 1 includes a media agent coordination mechanism 872, which analyzes the transaction by the media agent for that particular user. The media agent

coordination mechanism employs the dynamic profile developed from the transaction data in the transaction profile such that services may be offered to the user in a convenient fashion. The services, are not limited to music distribution but may also relate to video distribution, ordering of product from different service providers (both online as well as offline). Since the transaction profile includes not only events regarding financial transactions, and music listening patterns, but also sensory information.

The demographic profile information is information about the user that may or may not be confidential to that user. The user has the option to edit this information or even have this information deleted. However, if the data is deleted, the effectiveness of predicting services and products offered by way of the dynamic profile may be limited. The demographic aggregation and presentation mechanism 870 connects with the media agent coordination mechanism 872. The media agent coordination mechanism 872, extracts the demographic profile information associated with the user of that particular media agent, and handles this demographic information as a discrete "demographic unit". This demographic unit contains different attributes, such as user's age, profession, income and the like, but does not include personal identification information (name, social security number, credit card number). Accordingly, the demographic aggregation and presentation mechanism 870, is configured to combine different demographic units from different subscribers to the server system 1. Aggregating the different demographic units in this way, enables the server system 1 to present that demographic information in an anonymous, yet useful fashion to different service provider systems 860. For example, a candidate service provider system 860 may wish to pay a fee to the server system 1 in exchange for the server system 1 identifying different users having a certain demographic so that the service provider 860 may offer services for use by these demographic units.

For example, the demographic aggregation and presentation mechanism may be implemented as a web site that includes query features made by service provider system 860. The service provider system 860 may be interested in knowing how many demographic units in the server system 1 reflect people between the ages of 19 and 25, who enjoy watching football games, and enjoy eating pizza during those times. That particular demographic information number is presented graphically on the web page and the service provider system 860 may choose to dispatch, for a fee paid to the server system 1, an ECON (a combination

of e commerce and icon). The ECON associated with that particular service provider system 860 is extracted from the memory in the server system 1 and presented through the graphical user interface to the user of the media agent. In this way, the ECON, which may be a distinctive graphical mark associated with that service provider system 860, will introduce to that user the service offered by that service provider system 860. The ECON may be for a pizza delivery service or perhaps a service that offers streaming video of selected football games.

When the user of the media agent selects the ECON, a communication session is established between that particular agent and the service provider system 860, through an independent link on network 5 (not shown). Alternatively, the client-actuated link 858 may be used for handling the communications between the service provider system 860 and the media agent by way of the server system 1. The user of the media agent may then select and establish an agreement with the service provider system 860, to provide a particular service, as selected by the media agent. The transaction profile would then record this particular communication session between the media agent and the service provider system 860, as a discrete event for inclusion in the transaction profile.

By communicating this way, the real identity of the user of the media agent is not released to either the server system 1, or the service provider 860. Furthermore, the profile information used to define a demographic unit employed by the demographic aggregation and presentation mechanism 87 may be edited by the user of the media agent so that particular features associated with that particular user and his or her demographic profile may be amended. Thus the user remains in control of the content of the demographic information. Furthermore, it is the user that initiates the communication session by way of the user activated communication link 858, and not the server system 1, thus eliminating unwanted advertising or the like.

The user may maintain dominion and control over the user's demographic information, by use of the privacy assurance mechanism 852, which enables the user to not only limit the number of entities to which he can receive the profile information (by way of a digital signature), but may also provide a verification mechanism to verify that edited profile information has in fact been deleted if the user requests as much. Moreover, it uses the option to employ a different public key if the user chooses to no longer use an older key for

associating the user's profile with other user profiles. Because the server system 1, retrieves profile information from the media agent, the server system 1 relies on the media agent to provide a public key for the decrypting the profile, the user is assured that when the user no longer wishes to have others have access to the profile, the user may terminate that access by changing keys.

Figure 19 is directed to an embodiment which the profiles of different users, that were developed from different users may be exchanged or "swapped" between different users. As seen, a first client 3, initiates communications with a second client 7, by way of the network 5. A server system 1, is also connected to the network 5, and will help to verify that communications between the client's 3 and 7 are in fact "legitimate" communications. Moreover, suppose a user of client 7, wishes to receive the profile provided by the user of client 3. In this situation, the users of clients 7 to 3 may communicate with one another and agree that the user of client 3 will send a copy of the dynamic profile retained by the client 3, from the dynamic profile mechanism 856, to the client 7. In this situation, the transaction data developed about that particular user is sent by way of the network 5 to the client 7. The client 7 then combines that transaction data with transaction data of the user of client 7, so that the resultant dynamic profile used by the user at client 7, will reflect, a "merged" profile of two people.

Various people may wish to employ this profile sharing and combining approach, so as to learn more about one another prior to an initial meeting, such as a blind date. The dynamic profile combiner 882, includes options to either replace the profile 7, with the profile provided by client 3, or combine the two profiles. The combination may be a combination of all of the transaction data used to form the two profiles, or some subset thereof, as identified by the user of the client 7.

The situation may arise however, that the user of client 3, may choose to edit and thus alter the transaction data or the profile itself that is provided by the client 3, to the client 7. In this situation, the user of client 7 will be misled to believe that the profile provided by the client 3 is in fact indicative of the tendencies of the user of client 3. To avoid this authenticity uncertainty, the server system 1 includes a profile tracking mechanism 884. The profile tracking mechanism 884, in this embodiment, keeps a copy of the dynamic profile developed by the dynamic profile mechanism 856. In this way, if the profile editor 880 is

used by the user of the client 3, the user of the client 7, may verify with the server system 1, if the profile received from the user of client 3, was an un-altered profile. The profile certification mechanism 883, compares a copy of the profile sent from user of client 3, with the saved profile in the profile tracking mechanism 884. If the two profiles are the same, then the profile certification mechanism 886, dispatches a message to the user of the client 7, indicating that the profile sent by the user of client 3, is an unaltered profile. On the other hand, if the profile certification mechanism 886 determines that a change is present between the two profiles, then an indication is sent to the user of client 7, informing the client 7 that the change has been made.

The profile certification mechanism 886 optionally performs an exhaustive comparison, where it informs the user of client 7, of the magnitude of the differences between the two profiles. Accordingly, if the magnitude of the differences is not significant the difference may be the result of a transmission error, which would prompt the user of the client 7 to simply request that a new copy. On the other hand, if significant changes are made, the user of client 7 may have reason to believe that substantive changes were made to the profile by the user of client 3, and thus not opt to use the profile received from the user of client 3.

In part, the profile certification mechanism 886, may be used with the header lock mechanism discussed previously, so that profiles of certain celebrities will be "locked" and the user of the client 7, will be full confident that the profile received is from that particular celebrity.

PROFILE-INFLUENCED AND ANTICIPATORY AUDIO ADVERTISEMENT

Returning to Figures 17 and 18, the demographic aggregation and presentation mechanism 870, may coordinate with the media agent coordination mechanism 872 to determine when certain users have initiated communications with the server system 1. Under these circumstances, the profile associated with those users, may be analyzed by the media agent coordination mechanism 872, so as to identify, candidate consequent events, based on observed antecedent events for that particular user. These candidate consequent events, may be associated with services provided by one or more service provider systems 860. Under these circumstances, the media agent coordination mechanism 872 will trigger demographic

aggregation and presentation mechanism 870 to initiate a communication session with the service provider system 860 so as to inform the service provider system 860 that an opportunity has arisen to present a profile-influenced service message (perhaps in the form of an ECON) to the user of the media agent. Accordingly, the service provider system 860, may agree in the communication session that an ad should be launched. On the other hand, the service provider system 860 and a server system 1, may identify a set of rules under which the media agent coordination mechanism 872 determines that certain types of users, as indicated by the demographic units representing them, are available through the network 5. In this case, the server system 1 will itself serve as an agent for the service provider system 860, and present those services to that particular user in the form of a user-selectable ECON for example.

EFFICIENT SEARCH

Conventional search strategies and data structures are employed in conventional techniques that use nearest neighbor type searching. Exemplary conventional techniques, such as KD-trees and box-decomposition trees (or BD-trees for short) are discussed in *ANN Programming Manual* by David M. Mount (see, e.g. Appendix D, which is part of the present document).

The database at the server system will hold a huge number of signatures. These signatures are used for determining whether a particular signature provided by a client has in fact already been saved in the system server. The system server will keep this set of signatures therein, so that attributes associated with the particular signatures may be provided to different clients, when a client request information regarding the content associated with that particular signature.

While the signatures in the present embodiment contain 32 values, more generally, signatures may be used having N values. The efficient search mechanism according to the present invention, defines an N-dimensional space (one for each value in the signature), and each of the signatures are mapped into the signature space. As an example, a two-dimensional signature space is shown in Figure 20. A first dimension ("DIM 1") is shown to be orthogonal to a second dimension ("DIM 2"). Different signatures are shown as dots which fall within the space (i.e., bounding box). There is a Euclidian distance between

respective of the dots, where the efficient search mechanism will rely on the space so as to determine whether a particular signature (perhaps provided by a client) has already been saved in that signature space. The detection process is related to that employed in digital communications technology with regard to channel symbol detection, as discussed in *Sklar, B., Digital Communication Fundamentals and Applications*, Prentice Hall, 1988, ISBN 0-13-211939-0 025, the contents being incorporated herein by reference. One detection technique, however, is that if a Euclidian distance between the subject signature and a next-closest signature is less than the predetermined amount, then it is determined that the subject signature is the same as the other signature. If more than one candidate signature is within the predetermined distance from the subject signature, then the closest value is selected, unless side information is available that would indicate that the other candidate is in fact the correct candidate's signature. On the other hand, if no signatures are within the predetermined distance range, but at least one candidate falls within a second distance range, then if there is side information available that would suggest that candidate's signature is the same as the subject signature, then the candidate's signature is identified as being the subject signature. Other detection techniques may be employed as well.

Figure 21 shows a first step in a partitioning process, where the signature space is partitioned into subregions. For example, in the two-dimensional space of Figure 21, a median is selected along a dimension 1 (D1) such that half of the signatures within the boxed region (the overall region) fall on one side of the split in the dimension space, and the other half of the signatures fall on the other side thereof. The split is chosen in the dimension that has the largest span. Subsequently, an offset split ($D1_H$), is moved from the original split, along one of the dimensional axes, until that offset split comes into contact with the closest of the signatures in the subregion (R_1). Likewise, another offset split, is moved along the same dimensional axis toward the origin, until that offset split interacts with a first signature in that second subregion (R_2). In this way, the area for each of the subregions is reduced. This splitting operation continues by finding the next largest spread within each of the different regions (including subregions) and placing another split at the median (which in this case is D2) as seen in Figure 22. Then, offset splits are also prepared as illustrated by $D2_L$ and $D2_H$. As a consequence, a new route subregion R_2,1 and R_2,2, are found. This process repeats recursively until each of the regions or subregions are below a minimum spread in all

dimensions, or each region and subregion contains less then a minimum number of points.

Process steps associated with partitioning the space are described in the flowchart of Figure 27. The process begins in step S160, where a bounding box (space) is identified. The process then proceeds to step S161, where the largest spread along a dimension within that bounding box is identified. The process then proceeds to step S162, where a spread is cut at the median so that half of the signatures are on one side of the split and the other half of the signatures are on the other side of the split. The process then proceeds to step S163, where offset splits are shifted to the edges of the bounds for each of the subregions in step S163. Process steps S161-S163 are then recursively repeated in step S164 until each subregion contains less than a predetermined number of signatures, or each subregion is less than a predetermined spread in all dimensions. Subsequently the process ends.

A tree structure is then employed, for identifying a relationship between "neighboring regions". Neighboring regions are relevant to the search process because in order to identify closest distances between signatures, it is possible that a next-closest signature, is actually in a neighboring region. Thus, a tree structure such as that shown in Figure 23 is developed, where an ordered hierarchy is identified for identifying neighbors. The first node in the tree structure is a root node associated with an overall space. Once the root node is identified, the region is split as represented by two separate lines that extend from the root node. One of the nodes that is a child of the root node is identified as S1D1. S1D1, is initially a "neighbor" with the unlabeled node to the right. This neighboring relationship has a dotted line, however, this graphical representation, indicates that the neighboring relationship is terminated, as indicated by an "X" placed on the dotted line. The termination is done when the process further splits the region governed by node S1D1 into other subregions ($S1D1_L$ and $S1D1_H$). Moreover, S1D1 is no longer a "leaf node" (LN), meaning a terminating node in the tree structure.

Neighbors exist only for leaf nodes and thus parents of leaf nodes are not signified as having neighbors. Consequently, the originally recognized neighboring status between $S1D1_L$ and $S1D1_H$, as indicated by a dotted line, is subsequently destroyed (as signified by the X through the dotted line). As can be seen, leaf nodes extend from both $S1D1_L$ and $S1D1_H$. Neighboring statuses between the leaf nodes is indicated by dotted lines.

For each node, a data structure for information associated with that node, which will

facilitate the efficient search operation is described in Figure 24. Figure 24 includes subfields 890-897 to hold an identification of the node, a left child 891 and right child 892. A parent 893 of the node 890 is also indicated as part of the data structure. Furthermore, the dimension and point of the split for the children 894 is part of the data structure, as is the offset of the split for the node 895 as well as the list of neighbors and direction of the neighbors 896. Furthermore, the list of points contained in the particular region for that node 897 is saved as part of the data structure.

A process for determining whether a neighbor status exists between two leaf nodes is described in the flowchart of Figure 25. The process begins in step S120, where a nomenclature is defined. This nomenclature indicates that signatures X and Y are neighbors in dimension d if the nomenclature XY_d is used. X-->YZ signifies that Y and Z are children of X, and Y is on the low side of the partition (closer to the origin than Z in the given dimension). Under this nomenclature, the process proceeds to step S122 where an inquiry is made regarding whether AB_d and B-->CD and CD_d exists. If the condition for the inquiry in step S122 is affirmative, the process proceeds to step S124, where the condition AC_d XOR AD_d is identified and subsequently the process ends. However, if the response to the inquiry in step S122 is negative, the process proceeds to the inquiry in step S126, which inquires whether AD_d and B-->CD and CD_e and $e \neq d$. If the condition in step S126 is not meet, the process ends. However, if the response to the inquiry in step S126 is affirmative, the process proceeds to step S128, where a span of A in dimension d is found. Subsequently, the process proceeds to step S130, where the condition AC_d exists if any only if the partition (C,D) is greater than or equal to the lower bound of A in dimension d. The process then proceeds to step S132, where A and D are neighbors in the dimension d, if and only if the partition (C,D) is less than or equal to the upper bound in dimension d of region A. Subsequently, the process ends.

The efficient search process, given the above-described partitioning process and identification of neighbors, is described below with regard to Figure 26. The process begins in step S135, where the root node is identified. The process then proceeds to step S137, where the split dimension of a current node is identified and then the process proceeds to step S139, where the direction for that particular dimension and the signature is determined. The process then proceeds to step S141, where an inquiry is made regarding whether the resulting

node is a leaf node. If the response to the inquiry in step S141 is negative, the process returns to step S137. However, if the response to the inquiry in step S141 is affirmative, the process proceeds to step S143, where a distance is calculated for the new signature to be added to the data structure, for each of the signatures in the region for that leaf node. Identifiers are saved for each of the N-closest signatures in the regions in step S147. Subsequently, the process proceeds to step S149 where for each neighboring region, the distance to a boundary for that region is identified as being less than the distance to the Nth-closest so far of the signatures in the present region. If the response to the inquiry in step S149 is negative, the process ends, which means that all of the N-closest points to the new signature are found within the region for the leaf node. However, if the response to the inquiry in step S149 is affirmative, the N-closest signatures in the region for the leaf node are not necessarily found in the region for that leaf node, but may be found in one of the neighboring regions that has a boundary closer than a distance to the farthest of the N-nodes in the region for the leaf node. Accordingly, if the response to the inquiry in step S149 is affirmative, the process proceeds to step S151 where the distance is calculated to the new signature for each of the signatures in the neighboring region that had a distance to its boundary being less than the distance to the Nth-closest in the leaf nodes region. Subsequently, the process proceeds to step S153, wherein an identifier is saved for any signature in the neighboring region that is one of the N-closest. Consequently, after step S153, identifiers are saved that indicate the N-closest signatures to the new signature, regardless if those N-closest signatures are in the same region as the new signature, or are in a neighboring region. The process then ends.

Once the distances for the N-closest signatures are identified, a decision is then made regarding whether the distance from the new signature to the N-closest signatures is within a predetermined distance so as to conclude that the new signature equates with one of the existing signatures.

AUTOMATED MONITORING AND ADJUSTMENT OF VOLUME LEVEL

As can be seen in Figure 28, a waveform (e.g., one or more songs) to be processed is subdivided into discrete sections or "chunks". A section should be long enough so that it captures enough information about the underlying source data to provide the samples contained therein in an appropriate context, and it should be short enough so that the

likelihood of a large volume change within the section is relatively low. A length of 1/10th of a second has been found to be a suitable length for a section. An "ideal volume" is also defined, where the ideal volume is a volume that the adaptive volume technique according to the present invention will attempt to maintain. The purpose of maintaining the volume at the "ideal volume" is so that a user does not have to manipulate the volume to avoid annoying changes in dynamic range within a song. A maximum possible value for a 16-bit sample is 37,768. Because there is a possibility that samples could rise above the ideal volume in some cases, a value of 25,000 has been found to be a good value for the ideal volume. Of course other numbers and other ranges may be used as well.

One technique for adaptively adjusting volume is to first identify within each section of data, a sample that has the largest absolute value. Subsequently, the scale of the section may be changed by multiplying every sample by the ideal volume divided by that sections maximum absolute value. However, because the scheme discussed above normalizes each section independently, the change in the scaling between sections results in discontinuities that will be perceived as crackles in the audio stream. Therefore, the change in scaling is preferably changed gradually over the course of the section, perhaps linearly or non-linearly provided the rate of change is not abrupt.

In view of the above observation, an alternative process for controlling and maintaining the volume is to begin by, within each section of data, find the sample with the largest absolute value and save in memory a position of that sample within the section. The next step is then to scale the first sample in the section by the same value as the previous section was scaled. Subsequently, over the course of the section, the scale is gradually changed (perhaps in a linear fashion exponential, logarithmic, etc.) so that by the time the sample maximum occurs, the scaling has reached the point that the maximum sample will become the ideal volume when multiplied by the scale factor. Finally, the remaining samples are scaled in the section by the new scale factor and there is no further change in the scale factor for the remainder of the section.

Figure 28 illustrates two adjacent sections of an exemplary waveform. The line labeled "scale factor" reflects an amount by which samples are scaled although the actual scale factor is a number that is applied to the samples to have the peak portions of the waveform reach the ideal volume level. In section X+1, the scale factor is changed gradually

until the scale factor reaches the maximum sample. The scale factors then remains static for the remainder of the section.

There is a risk, however, that when the "straight-line" technique gradually changes the scale factor within a section, some of the samples may rise above the ideal volume. Figure 29 illustrates such a case. As can be seen, in the left hand side of section X+1, there is a sample that is not the maximum sample. However, using the gradually changing scale factor (shown to be a linear scale factor) based on the maximum sample would cause this sample to rise above the ideal volume. Therefore, the scale factor is changed more quickly then it would be were it based only on the maximum sample. This last case is illustrated in Figure 30.

Because a sudden change of volume between sections is probably significant and should not be negated by an automatic change in volume, a parameter is employed by a process according to the present invention to the system so that the total change in volume from one section to the next is always bounded relative to the earlier section. Stated more formally, if SF(X) is the scale factor of some section X, and if M is the maximum change between sections, then $|SF(X) - SF(X+1)| \leq M \times SF(X)$. If this condition is not true, then SF(X+1) is adjusted accordingly. An exemplary value of .25 were found to be a good value for M.

Finally, to avoid unbounded volume changes, there are maximum and minimal scale factors that should be employed. Moreover, the scale factor for any section should not exceed the maximum scale factor, and it may not be less than the minimum scale factor. Empirically, a maximum scale factor of 4 and a minimum scale factor or 25 were found to be suitable.

The mechanism for maintaining an adjustment of volume is performed in software, executed on the processor of Figure 2. Alternatively, hardware-embodiments of the volume control mechanism may be employed in an ASIC or PAL, for example.

ENHANCED ADVERTISING

Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views, and more particularly to Figure 31 thereof, a system for delivering and receiving commercials over a computer network includes one or more television and/or radio stations 1101, one or more streaming servers 1103 that receive

broadcast transmissions from the station 1101, an authentication and accounting server 1105, a database 1107, an ad server 1109, a database 1111, a database server 1113, a database 1115, clients 1117, 1121, and 1125, and a computer network such as the Internet 1129. As shown in Figure 31, the streaming server 1103, the authentication and accounting server 1105, the ad server 1109, the database server 1113, and the clients 1117, 1121, and 1125 are in communication with one another over the Internet 1129.

The station 1101 broadcasts programming to an audience of people. The programming includes commercials as well as programs, and thus, the broadcast signal on which the programming is carried includes both program segments when programs are aired and commercial segments when commercials are aired. Programs include any performance that is not a commercial, for example, news, weather, talk shows, and music. As used herein, the terms "advertisement" and "commercial" are used interchangeably to mean any paid announcement, for example, a promotion of goods or services. The streaming server 1103 receives the broadcast signal from the station 1101 via a satellite link or any other suitable device or network. For example, if the streaming server resides in the station 1101, then the streaming server 1103 could be physically connected to the soundboard in the station 1101.

The servers 1103, 1105, 1109, and 1113 are computers and/or software components. The servers 1103, 1105, 1109, and 1113 may be implemented on one or more computers, such as any suitable Dell, Compaq, Hewlett-Packard, and Sun Microsystems server models. Preferably, the servers 1103, 1105, 1109, and 1113 are programmed with software that is supported by UNIX or LINUX platforms.

The streaming server 1103 converts the broadcast signal received from the station 1101 into a data signal or stream that may be delivered over a computer network (e.g., the Internet 1129) and read by a remote computer (e.g., the client 1125). The streaming server 1103 stores network addresses, such as the Internet protocol (IP) address, of each computer receiving the data stream. The streaming server 1103 uses the network addresses to deliver the broadcasts to computers over the Internet 1129. The data stream may be formatted in any known manner, and preferably the streaming server includes multiple servers that generate the data signal in different formats (e.g., REALPLAYER, WINDOWS Media Player, MP3) to accommodate different hardware and software formats of the client computers used by the audience of the computer network broadcast. Redundancy and other known techniques may

be implemented as desired to increase bandwidth and/or manage network traffic.

Preferably, at least one streaming server 1103 is provided for each station 1101. Thus, when a user desires to listen to the broadcast from a particular station 1101, the user uses the media player running on his or her computer to connect to the streaming server 1103 corresponding to the station 1101 that the user wishes to receive. Further, the function of each streaming server 1103 may be duplicated on additional servers, using principles of redundancy and replication, to accommodate bandwidth and traffic.

The authentication and accounting server 1105 verifies that users have registered with the system by checking each user's username against a password for that username. The authentication and accounting server 1105 is connected to the database 1107, which stores records linking each username to a corresponding password. Accordingly, when a user accesses the system via a client (e.g., the client 1125), the user is first connected to the authentication and accounting server 1105. The authentication and accounting server 1105 may also register new users that have not previously established a username and password. As part of the registration process, the authentication and accounting server 1105 sends the user hypertext mark-up language (HTML) documents which are displayed on the user's computer (e.g., the client 1125) as HTML pages when the user's computer connects to the authentication and accounting server 1105. The HTML pages prompt the user to enter registration information, which is captured by the pages and transmitted to the authentication and accounting server 1105. The registration information includes a username and password for the user, demographic information, and any other desirable information of the user. The registration information may be stored in any of the databases 1107, 1111, and 1115.

The ad server 1109 receives requests for advertisements from the clients 1117, 1121, and 1125. The ad server 1109 is connected to the database 1111. In response to receiving a request for an advertisement from a client, the ad server 1109 looks up the requested ad in the database 1111 and delivers the requested ad to the requesting client. The database 1111 stores information for indexing computer network commercials and also stores files containing the computer network commercials to be sent to the clients 1117, 1121, and 1125.

The database server 1113 is connected to the database 1115 and processes requests to store information in and retrieve information from the database 1115. The database 1115 stores information that maps usernames with various groups and maps various groups to

advertisement identifiers (ad IDs). Additionally, the database 1115 stores event logging information for each user. The event logging information identifies the total length of time each user is logged into the system as well as the total length of time that the user listens to different broadcast shows and advertisements. The database server 1113 stores and accesses information in the database 1115.

The databases 1107, 1111, and 1115 are any suitable storage medium, including random access memory (RAM), magnetic tape, optical disks (e.g., DVDs or CDs), magneto-optical storage device, etc. Although three databases 1107, 1111, and 1115 are shown in Figure 31, the actual implementation of data storage may be varied as desired. For example, all of the information stored in the databases 1107, 1111, and 1115 could be stored in a single database, and/or any of the databases 1107, 1111, and 1115 could be divided into two or more databases to distribute the storage of information.

The clients 1117, 1121, and 1125 are hardware or software for providing an interface between the user and the servers 1103, 1105, 1109, and 1113 and for communicating with the servers 1103, 1105, 1109, and 1113. The clients 1117, 1121, and 1125 are implemented on personal computers, workstations, microcomputers, terminals, or any other suitable device (e.g. the computer system 1300 of Figure 43) for communicating with the servers 1103, 1105, 1109, and 1113 over a computer network, such as the Internet 1129. The clients 1117, 1121, and 1125 and the servers 1103, 1105, 1109, and 1113 communicate using any known protocol(s), such as TCP/IP and HTTP.

The clients 1117, 1121, and 1125 include respective media players (MPs) 1119, 1123, and 1127. The media players 1119, 1123, and 1127 are software modules that are downloaded off of the Internet (e.g., from a Web server and/or the authentication and accounting server 1105) or installed using appropriate installation software stored on a computer readable medium, such as a magnetic disk or CD-ROM, for example. According to the invention, the media players 1119, 1123, and 1127 store screen object information on the computer or device on which the media players 1119, 1123, and 1127 are running. The screen object information preferably includes graphical screen objects and information and/or data structures for mapping various graphical screen objects to ad IDs. The media players 1119, 1123, and 1127 cause one or more of the screen objects corresponding to a commercial to be displayed when the advertisement is played. For example, if a commercial for brand

ABC is playing on the client 1117, then the media player 1119 will cause company XXX's trademarked logo (i.e., the screen object) to be displayed to a user while the commercial for brand XXX is played. In a preferred embodiment, the screen objects are logos or logotypes corresponding to the service and/or product being advertised. Preferably, the screen objects are active links to Web sites of the advertiser, provider, retailer, and/or manufacturer of the advertised product or service.

Additionally, the media players 1119, 1123, and 1127 may include programming that enables the clients 1117, 1121, and 1125 to view HTML documents and/or receive programming broadcast from the servers 1103, 1105, 1109, and 1113, thereby eliminating the need for Web browser software on the clients 1117, 1121, and 1125. Alternatively, the media players 1119, 1123, and 1127 complement, or are extensions of, Web browser software. The media players 1119, 1123, and 1127 also recognize control signals received from the streaming server 1103, indicating that a commercial segment has begun in a broadcast. In response to receiving such signals, the media players 1119, 1123, and 1127 cause the clients 1117, 1121, and 1125 to connect to the ad server 1109 and request computer network commercials to be played on the clients 1117, 1121, and 1125. Preferably, ad IDs, corresponding to the computer network commercials, are stored in play list queues 1120, 1124, and 1128 in respective of the clients 1117, 1121, and 1125. The media players 1119, 1123, and 1127 may also include programming (e.g., pattern classification algorithms) for recognizing users' tastes in music and identifying music that particular users are likely to enjoy.

It is emphasized that the system of Figure 31 is for exemplary purposes only, as many variations of the hardware used to implement the present invention will be readily apparent to one having ordinary skill in the art. For example, the functionality of the ad server 1109 and the database server 1113 may be implemented on a single server. To implement these variations as well as other variations, a single computer (e.g., the computer system 1300 of Figure 43) may be programmed to perform the special purpose functions of two or more of any of the devices shown in Figure 31. On the other hand, two or more programmed computers may be substituted for any one of the devices shown in Figure 31.

The present invention stores information relating to usernames, passwords, demographic information, event logging information, and advertisements to be broadcast

over a computer network. This information is stored in one or more memories such as a hard disk, optical disk, magneto-optical disk, and/or RAM, for example. One or more databases, such as the databases 1107, 1111, and 1115, may store information used to implement the present invention. The databases 1107, 1111, and 1115 are organized using data structures (e.g., records, tables, arrays, fields, graphs, trees, and/or lists) contained in a memory such as a hard disk, floppy disk, optical disk, magneto-optical disk, and/or RAM, for example.

Figures 32-38 depict data structures used for broadcasting programs and computer network commercials to computers over a computer network. The data structures are depicted in a relational format, using tables, whereby information stored in one column (i.e., field) of a table is mapped or linked to information stored in the same row (i.e., record) in the other column(s) of the table. These data structures are used by the databases 1107, 1111, and 1115, the servers 1103, 1105, 1109, and 1113, and the clients 1117, 1121, and 1125 to store and retrieve information of the users of the Internet broadcast service as well as to coordinate the delivery of targeted advertisements and the display of screen objects to the users.

The data structures shown in Figures 32-38 are stored in the databases 1107, 1111, 1115, the clients 1117, 1121, and 1125, and/or any other suitable storage device(s). The design and implementation of various methods of database networking and Internet communications are described in Liu *et al.*, "Managing Internet Information Services," O'Reilly & Associates, Inc., 1994; Comer, "Internet Working with TCP/IP Volume I: Principles, Protocols, and Architecture," 2nd ed., Prentice-Hall, Inc., 1991; Comer and Stevens, "Internet Working with TCP/IP Volume II: Design, Implementation, and Internals," Prentice-Hall, Inc., 1991; Comer and Stevens, "Internet Working with TCP/IP Vol. III: Client-Server Programming and Applications," Prentice-Hall, Inc., 1993; Khoshafian *et al.*, "A Guide to Developing Client/Server SQL Applications," Morgan Kaufmann Publishers, Inc.; Hamilton *et al.*, "JDBC Database Access with Java, A Tutorial and Annotated Reference," Addison-Wesley Pub. Co., 1997; and Francis *et al.*, "Professional Active Server Pages 2.0," Wrox Press Ltd., 1998; each of which is incorporated by reference herein.

Figure 32 shows a password table 1201 that includes a field 1203 for storing usernames and a field 1205 for storing passwords. Thus, the password table maps each username to a password. The password table 1201 is preferably stored in the database 1107 connected to the authentication and accounting server 1105.

Figure 33 shows a group identification (ID) table 1301 that includes a field 1303 for storing usernames and a field 1305 for storing group identifiers (group IDs). Thus, the group ID table 1301 links each username in the field 1303 with a group ID in the field 1305. The group ID table 1301 is preferably stored in the database 1115 connected to the database server 1113. The group IDs stored in the field 1305 identify the classification of each user, based on information such as the user profile and demographic information of each user. For example, all male users over the age of 40 may be assigned a group ID of "1," and all female users over age 40 may be assigned a group ID of "2."

Figure 34 shows a user profile table 1401 that includes a field 1403 for storing usernames and a field 1405 for storing demographic information for each username in the field 1403. The field 1405 includes a field 1407 for storing the age of users, a field 1409 for storing the gender of users, a field 1411 for storing the zip code of users, a field 1413 for storing the profession of users, and a field 1415 for storing incomes of users. Additional fields may be added as desired to store additional demographic information of each user. Further, the user profile table 1401 may include information corresponding to the event logging information of the user. Preferably, the profession stored in field 1413 and the income store in field 1415 are discreet entries, generated by grouping the professions and/or incomes into ranges or classes. For example, the value of profession could be a "1," "2," "3," or "4" depending on whether each user classified himself as "unemployed," "retired," "teacher," or "accountant," respectively. Likewise, incomes may be represented by discreet values representative of a range of incomes (e.g., "1" represents an annual income between $20,001 and $30,000). The demographic information table 1401 is preferably stored in the database 1115 connected to the database server 1113. The demographic information table 1401 may be populated during a registration process in which the authentication and accounting server 1105 prompts a user for information with which to fill the demographic information table 1401.

Figure 35 shows an event logging table 1501 that includes a field 1503 for storing usernames and a field 1505 for storing event logging information. The field 1505 includes a field 1507 for storing a station address, a field 1509 for storing the time that a user's computer started receiving a broadcast from the station address in field 1507, and a field 1511 for storing the time that a user's computer ceased receiving a broadcast from the station address

stored in the field 1507. The station address stored in the field 1507 identifies the television or radio station that a user was listening to. The station address may be the IP address of the ad server or of the streaming server from which the station's broadcast was sent to the user's computer. Alternatively, the station address may be the call letters of the station sending the broadcast or any other suitable identifier. The event logging table 1501 is preferably stored in the database server 1113. The information or records stored in the event logging table 1501 may be generated by the respective clients and delivered to the database server 1113 to populate the event logging table 1501 periodically, such as when a user logs off.

Figure 36 shows an advertisement list table 1607 that includes a field 1609 for storing each group ID and a field 1611 for storing play lists of ad IDs. As noted above, the group IDs correspond to classifications, based on demographic information. Each ad ID stored in the field 1611 is a set of one or more alphanumeric identifiers identifying the advertisements to be delivered to users assigned to the group ID in the same row in the field 1609. Preferably, each ad ID is a 32-bit number. The curly brackets in the field 1611 indicate that one or more ad IDs exist for each group ID. The advertisement list table 1607 is preferably stored in the database 1115 connected to the database server 1113. The play lists stored in the field 1609 are sent to the clients where they are stored, for example, in corresponding play list queues 1120, 112, and 1128. In this embodiment, the play list queue of each client stores the ad IDs of the computer network commercials that the client requests from the ad server 1109.

Figure 37 shows an advertisement file table 1601 that includes a field 1603 for storing an ad ID for each computer network commercial. The advertisement file table 1601 also includes a field 1605 for storing the file location of each computer network commercial. Thus, each ad ID stored in the field 1603 corresponds to a file location stored in the field 1605. The file location indicates where the advertisement or commercial corresponding to the ad ID is stored. Preferably, the advertisement file table 1601 and the commercials are stored in the database 1114 connected to the ad server 1109.

Figure 38 shows a screen objects table 1701 that includes a field 1703 for storing ad IDs and a field 1705 for storing lists or sets of screen objects IDs. The curly brackets shown in the field 1705 indicate that one or more screen objects may be stored for each advertisement ID in the field 1703. The screen object IDs stored in the field 1705 identify a

memory location of a screen object or other image to be displayed to a user. Preferably, image files containing the screen objects are stored on the clients. These screen objects may be implemented as icons or icon-like images (e.g., logos) or other images (e.g., buttons) corresponding the manufacturer(s) or product(s) associated with the ad ID stored in the same row in the field 1703. Preferably, the screen objects table 1701 is stored on each of the clients 1117, 1121, and 1125. The screen objects table 1701 is populated when a user loads the media player and may be altered and/or updated whenever the user logs into the system via the authentication and accounting server 1105. The ad IDs stored in the field 1703 are the list of ad IDs stored in the field 1603. Thus, each ad ID (and corresponding computer network commercial) is associated with one ore more graphical screen objects stored in the client.

An example of the operation of the present invention and the interrelationship between the various hardware and software components of Figure 31 and the information stored in the data structures shown in Figures 32-38 will now be provided with reference to the flowcharts shown in Figures 39-42.

Figure 39 is a flowchart showing how a user registers with the Internet broadcast service provided by the system of Figure 31. For exemplary purposes, it is assumed that the user is using the client 1125 to access the Internet broadcast service.

In step 1901, the client 1125 loads the media player 1127. Additionally, the screen objects to be displayed on the client 1125 are stored in a suitable memory of the client 1125. Then, in step 1903 the user registers with the authentication and accounting server 1105. During registration, the user chooses, or alternatively receives, a username and password. Additionally, during the registration step 1903, the authentication and accounting server 1105 prompts the user to enter demographic information of the user as well as any other information to be used during the Internet broadcast of programs and advertisements. The user's user profile includes the information generated during the registration step 1903 and may also include other information of the user generated subsequent to the registration step 1903.

In step 1905, the user profile is stored in the user profile table 1401 and associated with the user's username. As a result, any of the user's demographic information stored in the user profile table 1401 may be accessed if the user's username is known.

In step 1907, the authentication and accounting server 1105 assigns the user to a group based on the user's user profile, including the demographic information of the user. Then, in step 1909, the user's username is associated with a group ID corresponding to the group assigned in step 1907. The user's username and group ID are stored in the group identification table 1301. Additionally, a user's user profile may be updated and/or the user may be reassigned to another group at a later time.

Figure 40 is a flowchart explaining how a broadcast is delivered to the user's computer (i.e., the client 1125) over the Internet 1129. In step 1911, the streaming server 1103 receives a broadcast signal from the radio station 1101. The broadcast signal may be received from the radio station 1101 via satellite or any other suitable connection.

In step 1913, the streaming server 1103 converts the broadcast signal to a suitable computer readable format, using the MP3 standard or any other suitable compression scheme. In step 1915, the user uses the media player 1127 to connect to the authentication and accounting server 1105 to log into the Internet broadcast system of Figure 31. During the log in step 1915, new versions or updates of the media player 1127 may be automatically or optionally downloaded. Also, additional or updated screen objects may be downloaded and stored on the client 1125. Further, the media player 1127 may send information about the graphical screen objects stored on the client 1125 back to the authentication and accounting server 1105 so that the authentication and accounting server 1105 may determine whether any of the screen objects stored on the client 1125 have been altered. This determination may be made by comparing the images stored on the client 1125 to the original images or by using header locking, which is described herein and in Ser. No. 60/174842 (attorney docket no. 10638-0011-2 PROV), entitled "METHOD, APPARATUS, AND COMPUTER PROGRAM PRODUCT FOR A MEDIA MANAGEMENT SYSTEM AND COMPONENTS THEREOF." This feature advantageously protects the trademarks of the advertisers by determining whether the advertiser images stored on the client 1125 have been altered or tampered with. Also, in step 1915, the user enters his or her username and password, which are sent to the authentication and accounting server 1105, to begin an Internet broadcast session.

In step 1917, the authentication and accounting server 1105 checks the username and password received from the client 1125 against the corresponding username and password in

the password table 1201 stored in the database 1107.

If the username and password received from the client 1125 are valid, then the process proceeds to step 1919. In step 1919, the database server 1113 uses the group ID table 1301 to associate the client 1125 with the group ID corresponding to the username received from the client 1125 in step 1915. Additionally, in step 1919 the database server 1113 begins to track event logging information of the user and stores this information in the event logging table 1501. The event logging information may be generated by either the client 1125 or the database server 1113, for example. Preferably, the event logging information is generated by the client 1125 and periodically sent to the database server 1113 where it is stored in the event logging information table 1501. However, any other suitable technique for tracking the event logging information may be implemented.

In step 1921, the database server 1113 uses the advertisement list table 1607 to identify the play list of ad IDs corresponding to the group ID associated with the client 1125 and sends the client 1125 the identified play list of ad IDs. In step 1923, the client 1125 stores the play list of ad IDs sent in step 1921. The play list of ad IDs is stored in the play list queue 1128, for example, and retrieved by the media player 1127 in the order in which the corresponding computer network commercials are requested from the ad server 1109.

In step 1925, the user selects a broadcast that he or she wishes to receive by using a graphical user interface (described below with reference to Figure 43) provided by the media player 1127. When the user selects the desired broadcast, the media player causes the client 1125 to connect to a streaming server that is providing the selected broadcast, in this case, the streaming server 1103. Preferably, the client 1125 or the authentication and accounting server 1105 automatically causes the client 1125 to connect to a streaming server that provides the broadcast in a format compatible with the audio and/or video compression software running on the client 1125.

In step 1927, the streaming server 1103 delivers the data signal carrying the selected broadcast to the client 1125. In step 1929, client 1125 receives the data signal delivered from the streaming server 1103. Then, in step 1931, the client 1125 reconstructs the program broadcast from the station 1101 and plays the reconstructed program to the user.

Figure 41 is a flowchart showing how the inventive Internet broadcasting system delivers computer network advertisements to the user. In step 1933, the streaming server

1103 sends a control signal to the client 1125. The control signal indicates the start of a commercial segment of the broadcast received from the station 1101. Since the timing of the commercial segments of broadcasts are known in advance of the actual broadcast, the streaming server 1103 may be programmed to send the control signal just before the start of the commercial segment. In step 1935, the client 1125 connects to the ad server 1109 in response to receiving the control signal from the streaming server 1103. Then, in step 1937 the client 1125 sends the first ad ID stored in the play list queue 1128 to the ad server 1109, which uses the advertisement list table 1601 to locate the corresponding computer network commercial in the database 1111. In step 1941, the ad server 1109 reads the corresponding computer network commercial out of the database 1111 and delivers the commercial to the client over the Internet 1129.

In step 1943, the client 1125 reconstructs and plays the commercial received from the ad server 1109. Also, in step 1943 the media player 1127 causes the client 1125 to display the screen object(s) corresponding to the commercial received from the ad server 1109 and corresponding to the ad ID sent to the ad server 1109 in step 1937. The screen objects table is used in step 1943 to identify and locate the graphical screen object(s) associated with the ad ID of the computer network commercial being played.

Next, in step 1945 the media player 1127 determines whether the play list queue 1128 contains any ad IDs remaining to be sent to the ad server 1109 in a request. If there are ad IDs remaining to be sent to the ad server 1109, then the process returns to step 1937 and the next ad ID is sent to the ad server 1109 in a request. If there are no ad IDs remaining to be sent to the ad server 1109, then the process proceeds to step 1947. In step 1947, the client 1125 reconnects to the streaming server 1103, which delivers the data signal corresponding to the broadcast signal to the client. As noted above, the timing of the commercial segments of the broadcast signal are known in advance. Therefore, the start, duration, and end of the computer network commercials (delivered in step 1941) may be timed to coincide with the commercial segments of the broadcast signal.

The present invention includes the computer screen interface and the associated programming used to generate the interface which is used for interaction with people who are associated with and carry out the operation of the invention. For example, the inputs of the invention are entered through the user interface of the screen and the outputs are displayed on

the screen, generated on printed paper, and/or played on speakers.

Figure 42 is an illustration of an exemplary network connection screen 1201 of the graphical user interface provided by the media players 1119, 1123, and 1127. The network connection screen 1201 includes a radio station field 1203, a buy button 1206, and a screen object field 1207, which are implemented as windows in the example of Figure 42. The radio station field 1203 includes a call letters field 1205 or other field for identifying the broadcast station currently being received by the client. If the user types in different call letters into the call letters field 1205, then the media player causes the client to connect to the streaming server delivering the selected broadcast. Alternatively, the radio station field 1203 may display a list of different broadcast stations, one of which may be selected by a user by pointing and clicking with a pointing device, such as a mouse. If a user wishes to purchase the advertised product or service, he or she may select the buy button 1206, which causes the client to guide the user through an appropriate sequence of step(s) for purchasing the advertised product or service. Various techniques for managing purchases over the Internet are well known.

The screen objects field 1207 displays screen objects, such as the screen object 1209, when computer network commercials are received by the client. Alternatively, the screen objects 1209 are not displayed in the screen objects field 1207, but are displayed elsewhere on the radio station interface screen 1201 or desktop. Preferably, the screen objects 1209 are graphical screen objects that correspond to the product or service currently being advertised on the client. Such graphical screen objects may take many forms, including icons or icon-like images, logos or logotypes, trademarks, and/or buttons, and may be displayed in various sizes. In a preferred embodiment, the screen objects 1209 are live links, which when selected by the user, cause the client to connect to a Web site of a manufacturer, retailer, product, etc. corresponding to the computer network commercial currently being played on the client.

All or a portion of the invention, including the processes set forth above, may be conveniently implemented using one or more conventional general purpose computers, microprocessors, and/or digital signal processors programmed according to the teachings of the present invention, as will be apparent to those skilled in the relevant art(s). Appropriate software can be readily prepared by programmers of ordinary skill based on the teachings of the present disclosure, as will be apparent to those skilled in the relevant art(s).

Figure 43 illustrates a computer system 1300 upon which an embodiment according to the present invention may be implemented. Computer system 1300 includes a bus 1302 or other communication mechanism for communicating information, and a processor 1304 coupled with bus 1302 for processing the information. Computer system 1300 also includes a main memory 1307, such as a random access memory (RAM) or other dynamic storage device (e.g., dynamic RAM (DRAM), static RAM (SRAM), synchronous DRAM (SDRAM), flash RAM), coupled to bus 1302 for storing information and instructions to be executed by processor 1304. In addition, main memory 1307 may be used for storing temporary variables or other intermediate information during execution of instructions to be executed by processor 1304. Computer system 1300 further includes a read only memory (ROM) 1309 or other static storage device (e.g., programmable ROM (PROM), erasable PROM (EPROM), and electrically erasable PROM (EEPROM)) coupled to bus 1302 for storing static information and instructions for processor 1304. A storage device 1311, such as a magnetic disk or optical disk, is provided and coupled to bus 1302 for storing information and instructions. The computer system 1300 may also include special purpose logic devices (e.g., application specific integrated circuits (ASICs)) or configurable logic devices (e.g., generic array of logic (GAL) or reprogrammable field programmable gate arrays (FPGAs)). Other removable media devices (e.g., a compact disc, a tape, and a removable magneto-optical media) or fixed, high density media drives, may be added to the computer system 1300 using an appropriate device bus (e.g., a small computer system interface (SCSI) bus, and enhanced integrated device electronics (IDE) bus, or an ultra-direct memory access (DMA) bus). The computer system 1300 may additionally include a compact disc reader, a compact disc reader-writer unit, or a compact disc juke box, each of which may be connected to the same device bus or another device bus.

Computer system 1300 may be coupled via bus 1302 to a display 1313, such as a cathode ray tube (CRT), for displaying information to a computer user. The display 1313 may be controlled by a display or graphics card. The computer system includes input devices, such as a keyboard 1315 and a cursor control 1317, for communicating information and command selections to processor 1304. The cursor control 1317, for example, is a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 1304 and for controlling cursor movement on the display

1313. In addition, a printer may provide printed listings of the data structure shown in Figures 32-38 or any other data stored and/or generated by the computer system 1300.

The computer system 1300 performs a portion or all of the processing steps of the invention in response to processor 1304 executing one or more sequences of one or more instructions contained in a memory, such as the main memory 1307. Such instructions may be read into the main memory 1307 from another computer-readable medium, such as storage device 1311. One or more processors in a multi-processing arrangement may also be employed to execute the sequences of instructions contained in main memory 1307. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions. Thus, embodiments are not limited to any specific combination of hardware circuitry and software.

As stated above, the system includes at least one computer readable medium or memory programmed according to the teachings of the invention and for containing data structures, tables, records, or other data described herein. Examples of computer readable media are compact discs, hard disks, floppy disks, tape, magneto-optical disks, PROMs (EPROM, EEPROM, flash EPROM), DRAM, SRAM, SDRAM, etc.

The term "computer-readable medium" as used herein refers to any medium that participates in providing instructions to processor 1304 for execution. A computer readable medium may take many forms, including but not limited to, non-volatile media, volatile media, and transmission media. Non-volatile media includes, for example, optical, magnetic disks, and magneto-optical disks, such as storage device 1311. Volatile media includes dynamic memory, such as main memory 1307. Transmission media includes coaxial cables, copper wire and fiber optics, including the wires that comprise bus 1302. Transmission media also may also take the form of acoustic or light waves, such as those generated during radio wave and infrared data communications.

Common forms of computer-readable media include, for example, hard disks, floppy disks, tape, magneto-optical disks, PROMs (EPROM, EEPROM, flash EPROM), DRAM, SRAM, SDRAM, or any other magnetic medium, compact disks (e.g., CD-ROM), or any other optical medium, punch cards, paper tape, or other physical medium with patterns of holes, a carrier wave (described below), or any other medium from which a computer can read.

Various forms of computer readable media may be involved in carrying one or more sequences of one or more instructions to processor 1304 for execution. For example, the instructions may initially be carried on a magnetic disk of a remote computer. The remote computer can load the instructions relating to the notification services to control call processing remotely into its dynamic memory and send the instructions over a telephone line using a modem. A modem local to computer system 1300 can receive the data on the telephone line and use an infrared transmitter to convert the data to an infrared signal. An infrared detector coupled to bus 1302 can receive the data carried in the infrared signal and place the data on bus 1302. Bus 1302 carries the data to main memory 1307, from which processor 1304 retrieves and executes the instructions. The instructions received by main memory 1307 may optionally be stored on storage device 1311 either before or after execution by processor 1304.

Computer system 1300 also includes a communication interface 1319 coupled to bus 1302. Communication interface 1319 provides a two-way data communication coupling to a network link 1321 that is connected to a local network 1323. For example, communication interface 1319 may be a network interface card to attach to any packet switched local area network (LAN). As another example, communication interface 1319 may be an asymmetrical digital subscriber line (ADSL) card, an integrated services digital network (ISDN) card or a modem to provide a data communication connection to a corresponding type of telephone line. Wireless links may also be implemented. In any such implementation, communication interface 1319 sends and receives electrical, electromagnetic or optical signals that carry digital data streams representing various types of information.

Network link 1321 typically provides data communication through one or more networks to other data devices. For example, network link 1321 may provide a connection through local network 1323 to a host computer 1325 or to data equipment operated by a service provider, which provides data communication services through an IP (Internet Protocol) network 1327 (e.g., the Internet 1129). LAN 1323 and IP network 1327 both use electrical, electromagnetic or optical signals that carry digital data streams. The signals through the various networks and the signals on network link 1321 and through communication interface 1319, which carry the digital data to and from computer system 1300, are exemplary forms of carrier waves transporting the information. Computer system

1300 can transmit notifications and receive data, including program code, through the network(s), network link 1321 and communication interface 1319.

REDUNDANT ARRAY OF INEXPENSIVE SERVERS

Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views.

As shown in Figure 45, the system according to the present invention includes one or more clone computers 4000 which serve as an interface point to the system for client computers 4004 through link L4007 to gain access to information maintained on the server system 4020. Each individual clone computer (e.g., 4005, 4006) is configured identically. Therefore, the clone computers 4000 component of the server system is scalable to meet performance requirements, and reliable since the failure of any single cloned computer will not cause the system as a whole to fail.

The clone computers 4000 are connected through a switch 4001 to a database 4002 and the server machines 4003. Each individual clone computer (e.g., 4005) maintains its own connection with the switch 4001, although Figure 45 shows only one exemplary link L4007. Both the database component 4002 and the server machines component 4003 include redundant elements thereby providing the performance and reliability that is an objective of the present invention. For example, the information maintained in database 4011 will be maintained in parallel in database 4012. Each individual database (e.g., 4011, and 4012) will maintain its own connection with the switch 4001, although Figure 45 shows only one exemplary link L4010. The server machines 4003 component of the system includes multiple inexpensive servers (e.g., 4013) each of which will maintain a connection with the switch 4001, although Figure 45 shows only one exemplary link L4009. Each individual server machine (e.g., 4013) is configured to have one or more storage devices (e.g, 4016) connected to it. As will discussed below, the information being maintained in the server system of the present invention is stored redundantly such that the failure of any individual storage device, or any individual server machine does not preclude the access of the information held on that device or that server elsewhere in the system.

The invention includes a software monitoring agent 4017 that is implemented as a process running on each clone computer (e.g, 4005), each database (e.g., 4011), and each

server machine (e.g, 4013), although other software architectures may be used as well. The software monitoring agent 4017 will detect and react to failures of components of the server system. When a component of the system fails, the software monitoring agent 4017 will cause the responsibilities that were maintained by that failed component to be redistributed to operational components of the server system. In this way, the server system of the present invention is a reliable and self-adjusting system with built-in failure detection and recovery capability.

The file load monitor 4018 is a software component of the server system that will report on which files and/or systems are being accessed within the server system most frequently. The file load monitor software 4018 will include the capability to make decisions regarding the redistribution of those files most frequently accessed thereby maintaining a consistent loading throughout all components of the server system. Again, any adjustments made to the system by the file load monitor software 4018 must be reflected in the database 4002. Moreover, the activities of the file load monitor software 4018 allow the server system of the present invention to be seamlessly scaled by adding additional inexpensive servers to the architecture. The file load monitor software 4018 will take advantage of any available resources in the server system to distribute the load.

When a client computer 4004 requests a file from the file server system 4020, that request is made to a clone computer (e.g., 4005). In the present example, the client computer will communicate with the clone computer shown as 4005 in Figure 45, and individual database 4011 will be accessed, but it should be recognized that the invention does not restrict which individual client computer 4004 can communicate with which individual clone computer 4000 or which individual database 4002 is accessed. The clone computer 4005 will then query the database (e.g., 4011) using the logical user-determined name for the file requested by the client. The database 4011 will return to the clone a list of the physical locations and file names corresponding to the locations where the requested file is stored within the server system 4020.

The clone computers 4000 each maintain a route table (e.g., 4019) that cross-references the device addresses corresponding to the physical locations of the file to the machine address of the server machine on which that device is mounted. Through the use of the route table (e.g., 4019), then, the clone computers 4000 can determine based on the

information received from the database which physical server machines 4003 are maintaining the requested file. By attaching to one of those server machines (e.g., 4013) containing the desired file, the clone computer 4000 can then access the requested file on the device (e.g., 4016) that is mounted on that server machine (e.g., 4013).

If the software monitoring agent 4017 detects the failure of a server machine 4003 it will cause the responsibilities of that machine to be redistributed within the system. For example, if a server machine were to be detected as inoperable, a "hot spare" (e.g., 4015) server machine may be instructed by the software monitoring agent 4017 to replace that failed server machine. By accessing the database 4002, the "hot spare" server machine (e.g., 4015) can determine which files were the responsibility of the failed server. The "hot spare" server machine (e.g., 4015) would be required to obtain copies of all those files, and store them on devices that were mounted to that "hot spare" server machine. Once all of the required files were stored on storage devices mounted on it, the "hot spare" server machine (e.g., 4015) must communicate to the clone computers 4000 that files being reported by the database as existing on the failed server machine storage devices, are now to be found on storage devices mounted to the "hot spare" server machine. The clone computers 4000 will then update their respective route tables (e.g., 4019) to reflect this change.

If the server system does not have a "hot spare" server machine available (e.g., 4015), the results of the above-described failure detection and recovery can be obtained through an alternative approach. If the software monitoring agent 4017 detects the failure of a server machine, and no "hot spare" server machine is available in the system, the software monitoring agent 4017 can cause the load of that failed server machine to be absorbed by the operational components of the server system. To achieve this, the software monitoring agent 4017 will be required to interact with the database 4002 to determine the load from the failed server machine that must be distributed, and must update the database to reflect the redistribution. In this example, it would be unnecessary to update the route tables (e.g., 4019) of the clone computers 4000.

As shown in Figure 46, the present invention abstracts the complexities of redundantly maintaining information in the server system 4020 into an intuitive file system representation. The view of the file system presented to the user 4100 is a typical file system view depicting file names and directories as determined by the user. As described above, the

server system of the present invention provides built-in redundancy of storage, has built-in failure detection and recovery, and has a built-in method for distributing the system load throughout the system. These capabilities may cause a particular file to be relocated within the server system 4020 for either failure recovery or performance reasons. Accordingly, the present inventor recognized the need for a scheme to hide the storage architecture complexities from the user. The server system of the present invention maintains the logical user view 4100 in the database (see Figure 45, 4002) together with a listing of the internal view of each physical location in the server architecture where the file is redundantly stored 4101. The internal view 4101 is maintained by the system and is hidden from the users accessing the system. It is the responsibility of the software monitoring agent (see Figure 1, 4017) to maintain the integrity of the information in the database.

As the examples above illustrate, the server system of the present invention provides a system that reacts to contingencies. It is therefore unnecessary, as with conventional approaches to scalable server systems, to reconfigure the system upon the event of a failure.

SINGLE-STEP MULTIHOST QUERYING

Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views.

As shown in Figure 47, a system according to the present invention includes a client computer 2700 connected through a secure transmission control protocol/Internet protocol (TCP/IP), as described in connection L2701 to a three layer architecture 2711. The secure link L2701 is implemented using the secure socket layer (SSL) protocol, although other encryption protocols may be used as well, and it will connect to an Internet protocol (IP) address that is shared by a network of interface computers 2702. Layer 1 of Figure 47 shows a network of interface computers. The network of interface computers 2702 connect through a TCP/IP connection L2703 to an IP address shared by a network of query translation computers 2704. Layer 2 in Figure 47 shows a network of query translation computers. The network of query translation computers 2704 maintains one or more TCP/IP connections to IP addresses shared within networks of server computers. In Figure 47, L2705 and L2706 are shown to indicate that each network of server computers will maintain a link to the network of query translation computers. The invention does not limit the number of clusters of server

computers that can be present in the architecture. Server clusters 2708 and 2709 are shown in Figure 47 for use in examples herein. The networks of server computers are shown as Layer 3 in Figure 47.

Each network of server computers, or cluster, is configured to respond to a particular class of request from the query translation computers 2704. For example, cluster 2708 may be configured to maintain information in a commercially available relational database management system, such as SYBASE. In this example, requests sent from the query translation computers 2704 through link L2705 to server cluster 2708, then, would be in a standard query language (SQL) format. Alternatively, server cluster 2709 may be configured to maintain information in a proprietary format. As such, requests sent from the query translation computers 2704 through link L2706 to server cluster 2709 would be in a proprietary format in order to perform the function of querying the information stored in a proprietary format. Thus, prior to forwarding a request from a Layer 2 query translation computer 2704 to a Layer 3 query server computer 2708, 2709 the query translation computer 2704 must determine, based on the class of information being requested by the client, which query server cluster 2708, 2709 should receive the request.

Each request sent from a query translation computer 2704 to a query server computer 2708, 2709 is sent using the multicast IP broadcasting protocol, although the broadcasting IP broadcasting protocol may also be used. This technique allows for a single request to be sent to a network of computers without regard to which of those computers may be capable of answering the request.

In another embodiment of the present invention, the query translation computers will expect to have each multicast message acknowledged from each of a predetermined set of query server computers that should have received the message. In order to accomplish this, it will be necessary for each query translation computer to maintain in memory information that will allow it to determine if all of the desired recipients of a given message have successfully received the message. If one or more of the predetermined set of query server computers does not acknowledge the multicast message, it will be assumed that the message was not received. The query translation computers will resend the message to those query server computers that did not acknowledge receipt using a unicast message.

Figure 48 is a flow diagram of a method performed according to the present invention.

The first step S2800 is to authenticate a connection L2701 by a client computer 2700 to an IP address that is shared by a network interface computers 2702 using the secure socket layer (SSL) protocol. Upon successful authentication at step S2800, the process proceeds to step S2801 where the SSL encrypted stream is decrypted into a real request. This step S2801 is performed by one of the network of interface computers 2702. Upon successful decryption at step S2801, the process proceeds to step S2802 where the decrypted real request is sent through a TCP/IP connection L2703 to an IP address shared by a network of query translation computers 2704. Upon receipt of the real request by one of the computers of the network query translation computers 2704, the process proceed to step S2803 where the request is unpacked to determine the nature of the request. Prior to fulfilling the request, the user ID and password contained in the request are authenticated at step S2804. This authentication step S2804 is performed by multicasting a query to the appropriate network of query server computers (i.e., 2708 or 2709) where user ID and password information are maintained. Upon successful authentication at step S2804, the process proceeds to step S2805 where the query translation computer will determine which substantive queries are requested and the appropriate destination for those requests. Once the queries are determined at step S2805, the process proceeds to step S2806 where those queries are issued to the appropriate networks of query server computers using a multicast technique. Each multicast query is sent to an IP address that is shared by a network of query server computers that maintain information that can be used to satisfy each particular request. Each multicast query issued at step S2806 is in a format consistent with the information maintained by the network of query server computers (i.e., 2708, 2709) to which the query was sent. In other words, one multicast request may be formatted as a standard query language (SQL) query whereas another multicast request may be formatted in a proprietary format to request information from a network of query server computers that maintains its information in a proprietary format. Since the query issued at step S2806 is multicast to the network of query server computers, each computer on that network configured to receive multicast messages will receive the request. Upon successful receipt of the multicast messages issued at step S2806, the process proceeds to step S2807 where only that server, or those servers, that can satisfy the request will respond to it. Upon receipt of a response from a query server computer at step S2808, the process proceeds to step S2809 where the query translation computer will pack a response

message to send back to the client. Once a response has been packed, the process proceeds to S2810 where the response is forwarded to an interface computer. The process then proceeds to step S2811 where the response is encrypted. Upon successful encryption of the response, the process proceeds to step S2812 where the response is forwarded back to the requesting client computer.

As the figures illustrate, and the above discussion describes, one application of this invention would be for a system that required secure connections with the clients, and where the information being requested was of a variety of formats. It should be clear, however, that there is no aspect of the present invention for querying multiple hosts with a single query that precludes its use for any system where multiple servers must be queried.

GRAPHICAL SKINNING FOR GRAPHICAL USER INTERFACES

Referring now to the drawings, wherein like reference numerals designate identical or corresponding parts throughout the several views.

Figure 50 shows a user image 2100, for a media player software application, where three regions of the image have been identified as control regions. The control regions identified correspond to a stop function 2101, a pause function 2102, and a play function 2103. It should be noted that the user was not restricted in the shape of the image, or in the shape of the control regions on the image corresponding to the software functions. As shown in Figure 50, the overall image is a rounded rectangle 2100, and the controls have corresponding regions defined by a circle 2101, a rectangle 2102, and a rounded rectangle 2103. Using the techniques of the invention, the user provides an image as shown in Figure 50 depicting the control areas in their default, or unselected, state.

Figure 51 shows an image as in Figure 50, but in this case, the control regions are shown in their active, or selected state. Note that the shapes and regions of Figure 51 correspond to the shapes and regions of Figure 50. In particular, the overall shape of the image in Figure 51, 2200, corresponds to the overall shape of the image in Figure 50, 2100. Furthermore, the shape and location of the control regions shown in Figure 51 correspond to the shape and location of the control regions in Figure 50.

Figure 52 shows the technique for mapping the software application functionality to the user image, or skin. Each function of the software application that is accessible thru the

user interface has been given a corresponding color code. These color codes, are defined by their red, green, and blue (RGB) values. It is well known to those of ordinary skill in the software art that each displayable color can be defined by its corresponding RGB value. The range of unique displayable colors runs from white (red=255, green=255, blue=255) to black (red=0, green=0, blue=0). By assigning each software application function a unique RGB value, the technique of the invention has greatly simplified the mechanism through which a user can define a region on an image to correspond to that function. For example, as shown in the color coded mask of Figure 52, each of the control regions has been given a unique RGB value (e.g., red=255, green=0, blue=0 corresponding to the red, or play control region 2303). Note that the color coded mask as shown in Figure 52 acts as a mask or overlay of the user image. In other words, in order to map the regions of the user image to the functions of the software application, that corresponding region must be painted with the appropriate color on the color coded mask, or overlay.

The inventor of the present invention has recognized that the creation of a color coded mask through the use of a common drawing tool is a technique that will be more readily acceptable to a computer user population with diverse backgrounds.

There are numerous advantages to this approach to skinning a software application. One advantage is that any user who can understand a simple drawing software application can employ those same techniques to put together a skin. All that is required for the user to understand to make a skin operational is the color coded values that correspond to the different functions of the software application. The technique of this invention imposes no arbitrary limitations such as the location or shape of the control regions. Furthermore, it should be noted, that using the technique of the invention, there is nothing to prevent the user from defining multiple regions of a given color. In other words, a user may define a skin for a media player software application that has multiple regions that correspond to the play function.

Figure 53 shows an additional use of the techniques of the invention in defining a skin for a software application. Figure 53 shows a menu animation mask that is used to specify the action of an animated menu system. As shown in Figure 53, the menu animation mask consists of pairs of color coded regions corresponding to the inactive and active positions to be occupied by the icons depicting the menu items. Figure 53 shows, as an example, a menu

animation mask for a system with seven menu items. Accordingly, seven pairs of color coded regions are shown. When the menu activation function is executed by the software application, the animation will cause the menu icons to traverse a path from the color coded start position for the menu icon to the color coded stop position for that same menu icon. Again, it should be noted that the skin creator need only understand which RGB color corresponds with which region, and each of the 14 color coded regions shown in Figure 53 will have a unique RGB value assigned to them.

The technique of the present invention can be employed to specify the configuration of any configurable capability of the software application for which the skin is being developed. For example, a mechanism for specifying the font and the spacing of the font used by the skin is shown in Figures 54 and 55.

Figures 54 and 5 show that by providing an image containing each of the displayable characters of a font, as in Figure 54, and another image containing a depiction of the width of each character in the font, as shown in Figure 55, the user has graphically provided enough information to allow the software application to display textual information using the font provided as if it were a true type font. This is just another example of how a software application can be configured by providing graphical images from which the software application can extract the configuration.

In order for the software application that is being skinned to configure itself, the software application digests the information contained in the graphical images provided by the skin. To accomplish this, a byte code compiler has been devised. The byte code compiler operates on the images defining the skin described above. Importantly, the byte code compiler is used not only to implement the graphical techniques to skin a software application described above, it also optimizes the display of the images of the skin defining the software application user interface.

Since the skinning technique of this invention is graphical in nature and uses color coding to define the areas of interests of the overlay mask, it is possible to define a color that corresponds to regions of the images that are not of interest to the application. This is significant from a performance prospective since those pixels of the images that are not of interest to the application need not be processed every time the display of the application is updated. Moreover, those pixels of information need not be kept in memory by the software

application, thereby reducing the memory required to store the images making up the skin.

Figure 56 shows a structure created by the byte code compiler that is used to hold the information pertaining to the images and behavior of the skin, as well as the machine code used by the software application to display the images of the skin. In compiling the images defining the skin, the byte code compiler generates machine code used to display the images of the skin corresponding to the state of the software application. The byte code compiler generates a block of information, shown in Figure 56, including header information 2400 that describes the information held in the information block such as pointers to the executable code contained therein; the images and icons themselves 2401, 2402, and 2403; clipping regions used for animations 2404; font display information 2405; and machine executable code used to draw the images of the skin 2406. The software application for which the skin was developed can load this block of information into memory and access it at run time to invoke the appropriate behavior.

The skinable software application using the present invention is written so as to recognize the presence of the compiled block of information for the skin. If the application does not sense the presence of such a block, the application will cause the byte code compiler to compile a block from the files containing the images defining the skin for that application. In this way, the user need not be concerned with how to compile a set of images into an operational skin.

The use of the color coded masks as shown in Figures 52 and 53 allow the byte code compiler to optimize with respect to memory requirements. To illustrate this, note that the only differences between Figure 50 and Figure 51 are contained in the regions corresponding to the control regions. Consequently, it is not necessary for the application to store the entirety of Figure 51, but rather, the application need only store the information in Figure 51 corresponding to the control regions (i.e. 2201, 2202, 2203). The byte code compiler can therefore optimize by creating a block of executable code to display the pixels corresponding to the play region 2203 when the media player software application senses a select command from a pointing device over that region. This approach is efficient not only from a memory storage prospective, but also from a screen refresh prospective.

Figure 57 is a flow diagram showing the steps taken in one embodiment of the present invention to create a graphical user interface skin for a software application. The first step

S2500 is to create or obtain computer readable graphical images that will be used to define the look and feel of the software application. Once the images are obtained, the process proceeds to step S2501 where the regions on those images that will correspond to the functions provided by the software application are identified. The user is not limited with respect to size, shape, or location of the control regions being defined on the images. Once the control regions have been identified, the process proceeds to step S2502 where a color coded mask overlay is created by painting the control regions with the appropriate color corresponding to the colors specified by the software application. Upon completion of step S2502 the user will have a plurality of computer readable images containing the look and feel of the skin as well as an identification of the control areas on those images. The inventor of the present invention has recognized that contained in this group of images is enough information from which a functional skin can be created.

Figure 58 is a flow diagram of one embodiment of the present invention showing how the information making up the skin is utilized by the software application to present the functionality of that application to the user. The process begins at step S2600 by the user executing the software application providing information identifying which skin the software application should use. The process then proceeds to step S2601 where the skinable software application will check to verify that the information pertaining to the skin to be used by the application is present. If that information is not present (i.e., a "NO" answer at step S2601), the process proceeds to step S2603 where the software application executable will cause the byte code compiler to execute on a group of images defining a skin stored in a predetermined location. The result of this compilation at step S2603 is a block of information and machine code that is required by the software application executable to make use of the skin. Once that information is present (i.e., successful completion of step S2603, or alternatively a "YES" answer at step S2601), the process proceeds to step S602 where that information will be loaded into memory by the software application. Once the skin information generated by the byte code compiler has been loaded into memory at step S2602, the process proceeds to step S2604 where the executable software application will access that information and machine code during the execution of the application.

As the figures illustrate, and the above discussion suggests, one useful application of this invention would be for a skinable media player software application. It should be clear,

however, that there is no aspect of the present invention for graphically skinning software applications that precludes its use for any software application with a graphical user interface. The concept for skinning an application so as to provide a custom user interface, is a common one among computer professionals. The present invention enables those with a limited computer literacy to take advantage of this heretofore advanced concept.

The processes set forth in the present description may be implemented using one or more conventional general purpose microprocessors and/or digital signal processors programmed according to the teachings of the present specification, as will be appreciated to those skilled in the relevant art(s). Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will also be apparent to those skilled in the relevant art(s).

The present invention thus also includes a computer-program product which may be hosted on a storage medium and include instructions which can be used to program a computer to perform a process in accordance with the present invention. The storage medium can include, but is not limited to, any type of disk including floppy disk, optical disk, CD-ROMS, and magneto-optical disks, ROMS, RAMs, EPROMs, EEPROMs, flash memory, magnetic or optical cards, or any type of media suitable for storing electronic instructions.

Obviously, numerous modifications and variations of the present invention are possible in light of the above teachings. It is therefore to be understood that within the scope of the appended claims, the invention may be practiced otherwise than as specifically described herein.

CLAIMS:

The invention claimed is:

1. A media management system, comprising:
   (a) a server connected to a network wherein said network supports communication between said server and at least one client processor;
   (b) said server to respond to commands from at least one of said client processors and supply digital data to said client processor through said network;
   (c) at least one client processor including a decoder to decode encoded discrete digital data messages, an evaluator to create a value corresponding to the digital data, and an assignor that associates a value to digital data; and,
   (d) a transceiver controlled by said client processor that sends and receives messages from the network for evaluation by said client processor.
2. A media management system as claimed in claim 1, wherein said client processor and transceiver are a personal computer.
3. A media management system as claimed in claim 1, wherein said client processor and transceiver are a wireless telephone.
4. A media management system as claimed in claim 1 that further comprises a profiler for predicting future events based on the activities of a user.
5. A media management system as claimed in claim 4 where in said profiler further includes an associator to coordinate digital data with a user.
6. A media management system, comprising:
   (a) a media data player;
   (b) a profile application means for evaluating a profile;
   (c) an analog signature characterization mechanism for creating analog signatures of media data;
   (d) a dynamic profiling means to develop a profile wherein said dynamic profiling means shares data with said profile application means and evaluates the values of analog signatures; and,
   (e) a header locking means to maintain the integrity of content;
7. The media management system of claim 6, wherein said header locking means further prevents tampering with media data without detection.

8. The media management system of claim 6 further comprising an item set chooser means to direct recommendations of media data.

9. The media management system of claim 6 further comprising media agent means to enable the control of demographic profile information.

10. The media management system of claim 6 further comprising automated monitoring means to adjust the volume levels of a media data.

11. A media management system, comprising:

(a) a server;

(b) at least one client computer;

(c) a plurality of clone computers that interface said client computer to said server;

(d) a database means for storing data; and

(e) a software monitoring means to detect and react to failures of components of the sever system wherein said software monitoring means includes redistribution means to adjust data to operational components of the media management system.

12. The media management system of claim 11 wherein said software monitoring means includes file load monitor means to track and report the usage of files.

13. The media management system of claim 12 wherein said file load monitor means includes a controller to distribute data according to load evaluation.

14. The media management system of claim 11 wherein each of said clone computers comprises a route table.

15. A media input/output device, comprising:

(a) an input that is electrically coupled to a channel;

(b) a storage that receives digital data from the input;

(c) the storage having capacity to maintain digital data for timed retrieval by processor;

(d) a processor that is enabled to retrieve data from storage and includes an examiner that calculates a value of said digital data wherein said examiner enables the confirmation of the integrity of said digital data;

(e) said processor controlling the location of said digital data; and

(f) an output coupled to said processor or storage for converting said digital data in human perceptible form.

16. The media input/output device claimed in claim 15 wherein the processor further comprises an integrator that stores said value within said digital data.

17. A media management system, comprising;

(a) analog signature means for the creation of analog signatures of media data;

(b) means to transmit said analog signature; and

(c) signature database means for storing and evaluating said analog signatures. system, method, apparatus and/or computer program product, that includes a volume maintenance mean for adjusting the power previously discussed used either individually or in combination with other constituent components discussed herein.

18. A database management system, comprising:

(a) a collection of analog signatures; and

(b) an efficient search means for managing analog signatures.

19. A method for delivering commercials to a computer, comprising the steps of:

(a) receiving a broadcast signal sent from a broadcast station, the broadcast signal including commercial segments when broadcast commercials are broadcast and program segments when no broadcast commercials are broadcast;

(b) converting the broadcast signal to a first data signal;

(c) delivering the first data signal over a computer network to a computer during the program segments of the broadcast signal;

(d) generating a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials; and

(e) delivering the second data signal over the computer network to the computer during the commercial segments of the broadcast signal so that the computer receives the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

20. The method of claim 19, wherein the broadcast signal is one of a television signal and a radio signal.

21. The method of claim 19, wherein the first data signal and the second data signal comprise packets of data and the computer network comprises a packet switching network.

22. The method of claim 19, further comprising the step of:
sending a control signal to the computer to cause the computer to receive the second data signal instead of the first data signal during the commercial segments of the broadcast signal.

23. The method of claim 19, further comprising the step of:
(a) associating the computer with a user group;
(b) associating the computer network commercials with the user group; and
(c) selecting the computer network commercials on the basis of the user group.

24. The method of claim 19, wherein the computer network commercials comprise:
advertisements targeted to a user of the computer.

25. The method of claim 20, further comprising the steps of:
assigning the user to the user group based on a user profile of the user.

26. The method of claim 19, wherein the user profile comprises:
demographic information of the user.

27. The method of claim 26, wherein the demographic information comprises at least one of: age, gender, zip code, profession, and income.

28. The method of claim 23, further comprising the step of:
tracking event logging information corresponding to the user's use of the computer, the user profile including the event logging information.

29. The method of claim 28, wherein the usage information comprises:
(a) first time information corresponding to the length of time that the computer receives the first data signal; and
(b) second time information corresponding to the length of time that the computer receives the second data signal.

30. A method for playing network commercials on a computer, comprising the steps of:
(a) receiving at a computer a first data signal, generated from broadcast signal sent from a broadcast station and including commercial segments when

broadcast commercials are broadcast and program segments when a program is broadcast, during the program segments of the broadcast signal;

(b) reconstructing the program from the first data signal and playing the program on the computer during the program segments of the broadcast signal;

(c) receiving at the computer a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials, during the commercial segments of the broadcast signal; and

(d) reconstructing the computer network commercials from the second data signal and playing the computer network commercials on the computer during the commercial segments of the broadcast signal so that the computer plays the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

31. The method of claim 30, further comprising:
displaying, while each computer network commercial is played, a screen object associated with the computer network commercial being played

32. The method of claim 31, wherein each screen object corresponds to an advertiser corresponding to the computer network commercial being played.

33. The method of claim 32, wherein the each screen object is a logo of the corresponding advertiser.

34. The method of any of claims 31-33, wherein the computer screen objects are icons or icon-like images.

35. The method of any of claims 31-33, wherein each screen object is an active link to a network site of the corresponding advertiser.

36. The method of any of claims 31-35, wherein at least some of the screen objects are logos corresponding to respective of the respective advertisers.

37. A system for delivering commercials to a computer, comprising:

(a) means for receiving a broadcast signal sent from a broadcast station, the broadcast signal including commercial segments when broadcast commercials are broadcast and program segments when no broadcast commercials are broadcast;

(b) means for converting the broadcast signal to a first data signal;

(c) means for delivering the first data signal over a computer network to a computer during the program segments of the broadcast signal;

(d) means for generating a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials; and

(e) means for delivering the second data signal over the computer network to the computer during the commercial segments of the broadcast signal so that the computer receives the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

38. A system for delivering commercials to a computer, comprising:

(a) a broadcast server configured to receive a broadcast signal sent from a broadcast station, the broadcast signal including commercial segments when broadcast commercials are broadcast and program segments when no broadcast commercials are broadcast; convert the broadcast signal to a first data signal; deliver the first data signal over a computer network to a computer during the program segments of the broadcast signal;

(b) an ad server configured to generate a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials; and deliver the second data signal over the computer network to the computer during the commercial segments of the broadcast signal so that the computer receives the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

39. The system of claim 38, wherein the broadcast signal is one of a television signal and a radio signal.

40. The system of claim 38, wherein the broadcast server comprises:

(a) a streaming server configured to stream the first data signal over the computer network to the computer; and

(b) wherein the first data signal and the second data signal comprise packets of data and the computer network comprises a packet switching network.

41. The system of claim 38, wherein the broadcast server is further configured to send a

control signal to the computer to cause the computer to receive the second data signal instead of the first data signal during the commercial segments of the broadcast signal.

42. The system of claim 38, further comprising:

(a) at least one memory configured to associate the computer with a user group, associate the computer network commercials with the user group; and

(b) a database server configured to select the computer network commercials on the basis of the user group.

43. The system of claim 38, further comprising:

(a) a database server configured to track event logging information corresponding to the user's use of the computer, the user profile including the event logging information.

44. The system of claim C, wherein the usage information comprises:

(a) first time information corresponding to the length of time that the computer receives the first data signal; and

(b) second time information corresponding to the length of time that the computer receives the second data signal.

45. A system for playing network commercials, comprising:

(a) means for receiving at a computer a first data signal, generated from broadcast signal sent from a broadcast station and including commercial segments when broadcast commercials are broadcast and program segments when a program is broadcast, during the program segments of the broadcast signal;

(b) means for reconstructing the program from the first data signal and playing the program on the computer during the program segments of the broadcast signal;

(c) means for receiving at the computer a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials, during the commercial segments of the broadcast signal; and

(d) means for reconstructing the computer network commercials from the second data signal and playing the computer network commercials on the computer

during the commercial segments of the broadcast signal so that the computer plays the computer network commercials instead of the broadcast commercials during the commercial segments of the broadcast signal.

46. A system for playing network commercials on a computer, comprising:

(a) means for receiving a first data signal, generated from broadcast signal sent from a broadcast station and including commercial segments when broadcast commercials are broadcast and program segments when a program is broadcast, during the program segments of the broadcast signal, and for receiving a second data signal corresponding to computer network commercials that are at least partially different from the broadcast commercials, during the commercial segments of the broadcast signal;

(b) a processor configured to receive the first and second data signals from the means for receiving, reconstruct the program from the first data signal, and to reconstruct the computer network commercials from the second data signal; and

(c) at least one output device configured to play the program and the computer network commercials, the processor controlling the at least one output device to cause the at least one output device to play the program during the program segments of the broadcast signal and to play the computer network commercials during the during the commercial segments of the broadcast signal so that the computer network commercials are played instead of the broadcast commercials during the commercial segments of the broadcast signal.

47. A memory for storing information for delivering commercials to a computer, comprising a data structure including:

(a) a first field for storing usernames corresponding to users;

(b) a second field for storing a group identifier for each username in the first field, each group identifier corresponding to a classification of the corresponding user based, at least in part, on demographic information of the user;

(c) a third field for storing groups of advertisement identifiers corresponding to advertisements to be delivered over a computer network to the users during

commercial segments of a network broadcast, each group of advertisements identifiers corresponding to one of the group identifiers.

48. A memory for storing information for delivering commercials to a computer, comprising a data structure including:

(a) a first field for storing advertisement identifiers to be sent to a server from a computer and corresponding to advertisements requested over a computer network by the computer during commercial segments of a network broadcast; and

(b) a second field for storing screen objects associated with respective of the advertisement identifiers and to be displayed on the computer, each screen object being displayed in the computer when the advertisement corresponding to the associated advertisement identifier is received by the computer from the server.

49. A system of providing redundant storage for information, comprising:

(a) a database populated with data mapping a plurality of physical locations of a plurality of files being stored to a user provided name of said plurality of files;

(b) at least one network computer configured to store said plurality of files;

(c) at least one clone computer configured to,

receive a request from a client computer for said user provided name of said plurality of files,

query said database to retrieve said plurality of physical locations of said plurality of files,

return said plurality of files corresponding to said user provided name contained in said request from said client computer.

50. A method of querying a plurality of servers with a single query comprising the steps of:

(a) creating a network to connect said plurality of servers;

(b) configuring said network with a shared IP address that is accessible to each member of said network of said plurality of servers;

(c) configuring at least one said member of said network to respond to messages conforming to a multicast protocol; and

(d) communicating said single query using said multicast protocol to said shared IP address of said network of said plurality of servers.

51. A method of skinning a graphical user interface of a software application comprising the steps of:

(a) obtaining at least one computer readable interface image;

(b) defining at least one control region on said at least one computer readable interface image;

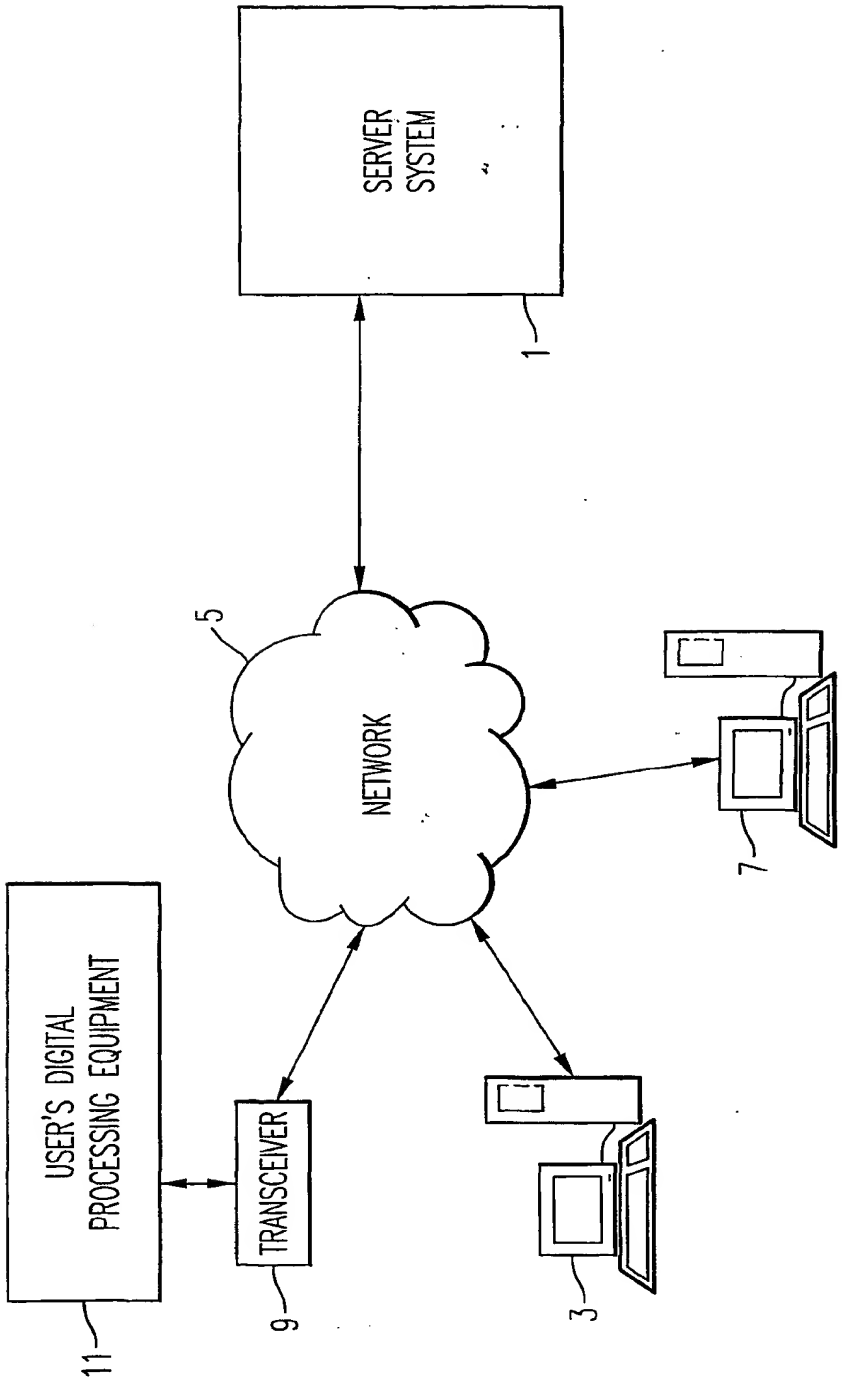
(c) creating at least one computer readable mask overlay image such that said at least one computer readable mask overlay image corresponds in size and shape to said at least one computer readable interface image;

(d) defining at least one mask overlay region on said at least one computer readable mask overlay image such that said at least one mask overlay region corresponds to said at least one control region;

(e) compiling said at least one computer readable interface image and said at least one computer readable mask overlay image; and

(f) accessing the information generated by said compiling step with said software application such that said software application functions are performed by interfacing with said at least one control region of displayed said at least one computer readable interface image.

FIG. 1



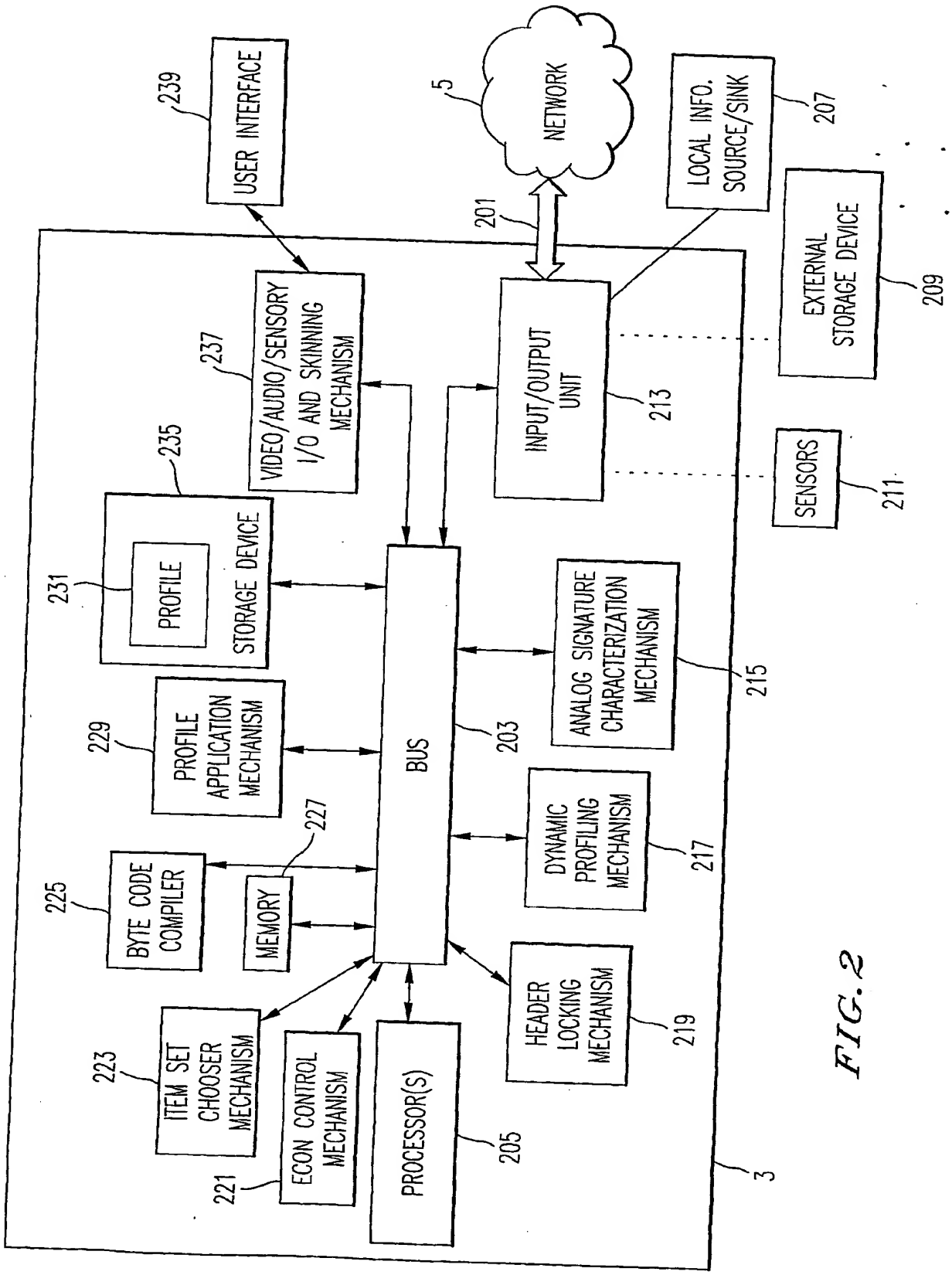

239
USER INTERFACE
5
NETWORK
LOCAL INFO. SOURCE/SINK
207
201
EXTERNAL STORAGE DEVICE
209
237
VIDEO/AUDIO/SENSORY I/O AND SKINNING MECHANISM
INPUT/OUTPUT UNIT
213
235
231
PROFILE
STORAGE DEVICE
SENSORS
211
229
PROFILE APPLICATION MECHANISM
BUS
203
ANALOG SIGNATURE CHARACTERIZATION MECHANISM
215
227
225
BYTE CODE COMPILER
MEMORY
DYNAMIC PROFILING MECHANISM
217
223
ITEM SET CHOOSER MECHANISM
HEADER LOCKING MECHANISM
219
221
ECON CONTROL MECHANISM
PROCESSOR(S)
205
3

FIG. 2

FIG. 3

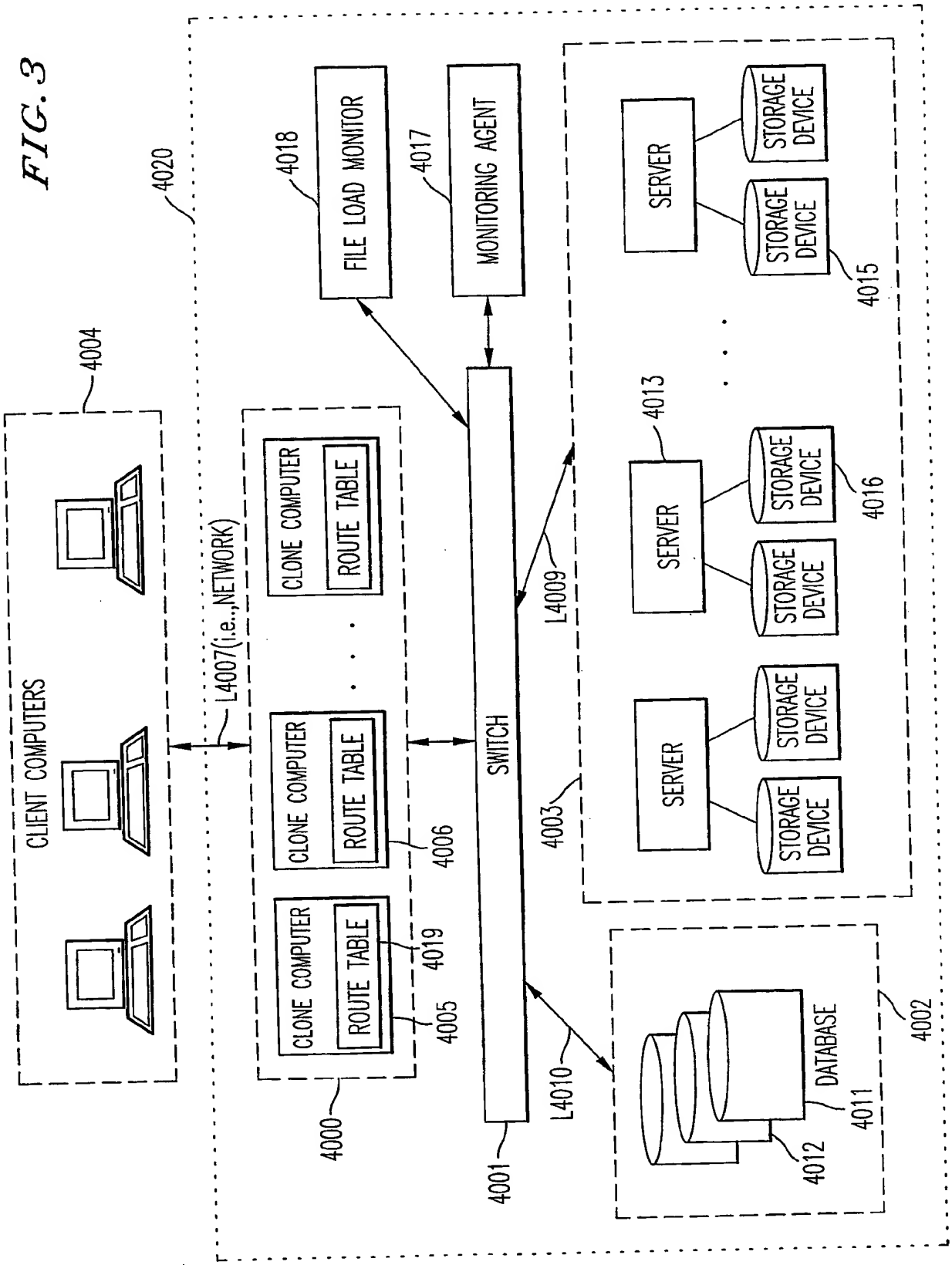
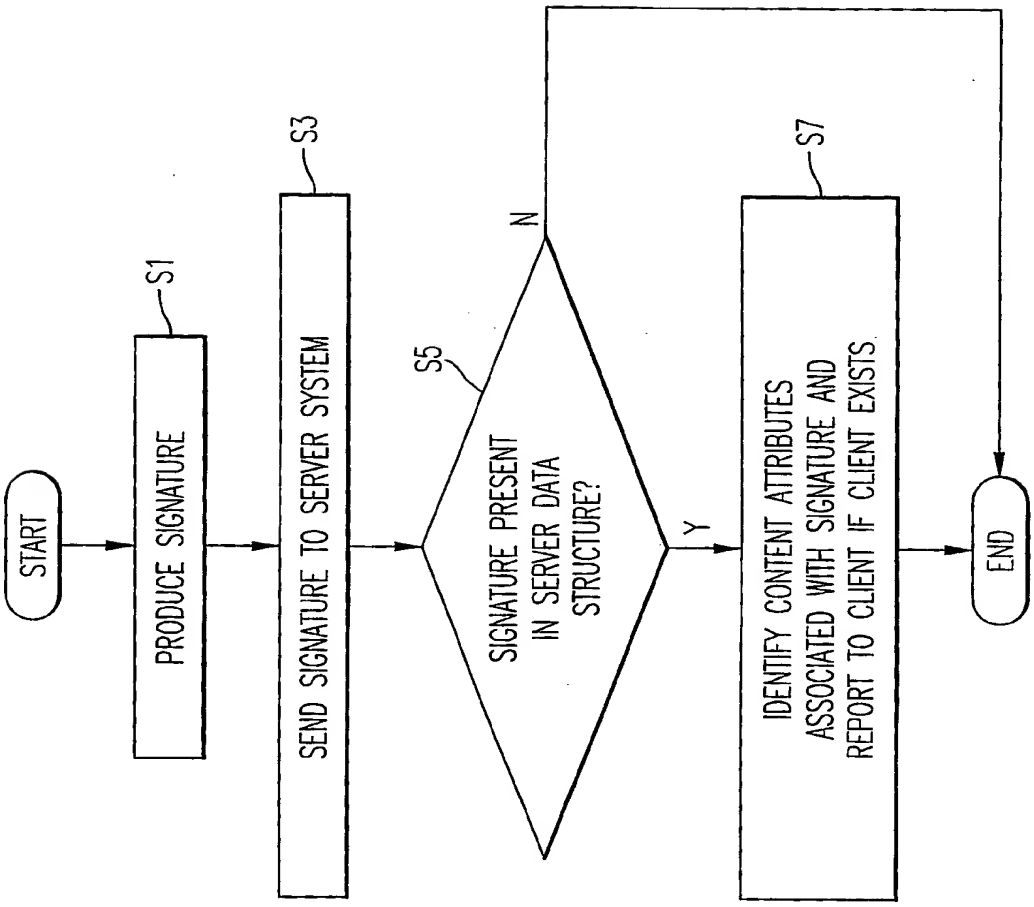

4020
4018
FILE LOAD MONITOR
4017
MONITORING AGENT
4004
CLIENT COMPUTERS
L4007(i.e.,NETWORK)
CLONE COMPUTER
ROUTE TABLE
4006
4019
4005
4000
SWITCH
4001
L4009
L4010
4003
SERVER
STORAGE DEVICE
4013
4016
4015
4002
DATABASE
4011
4012

FIG. 4


START
PRODUCE SIGNATURE
S1
SEND SIGNATURE TO SERVER SYSTEM
S3
SIGNATURE PRESENT IN SERVER DATA STRUCTURE?
S5
N
Y
IDENTIFY CONTENT ATTRIBUTES ASSOCIATED WITH SIGNATURE AND REPORT TO CLIENT IF CLIENT EXISTS
S7
END

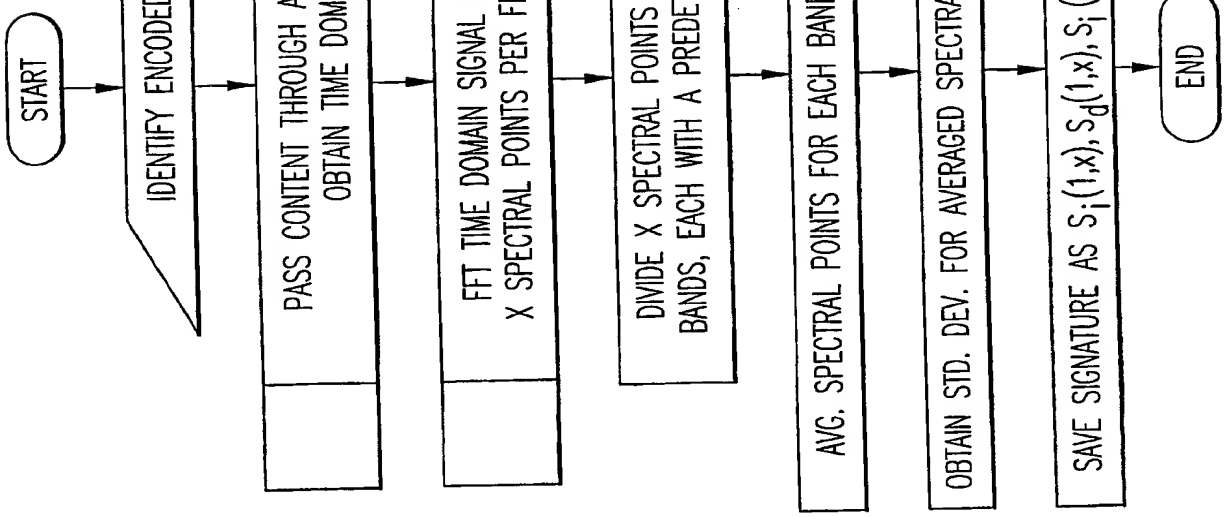
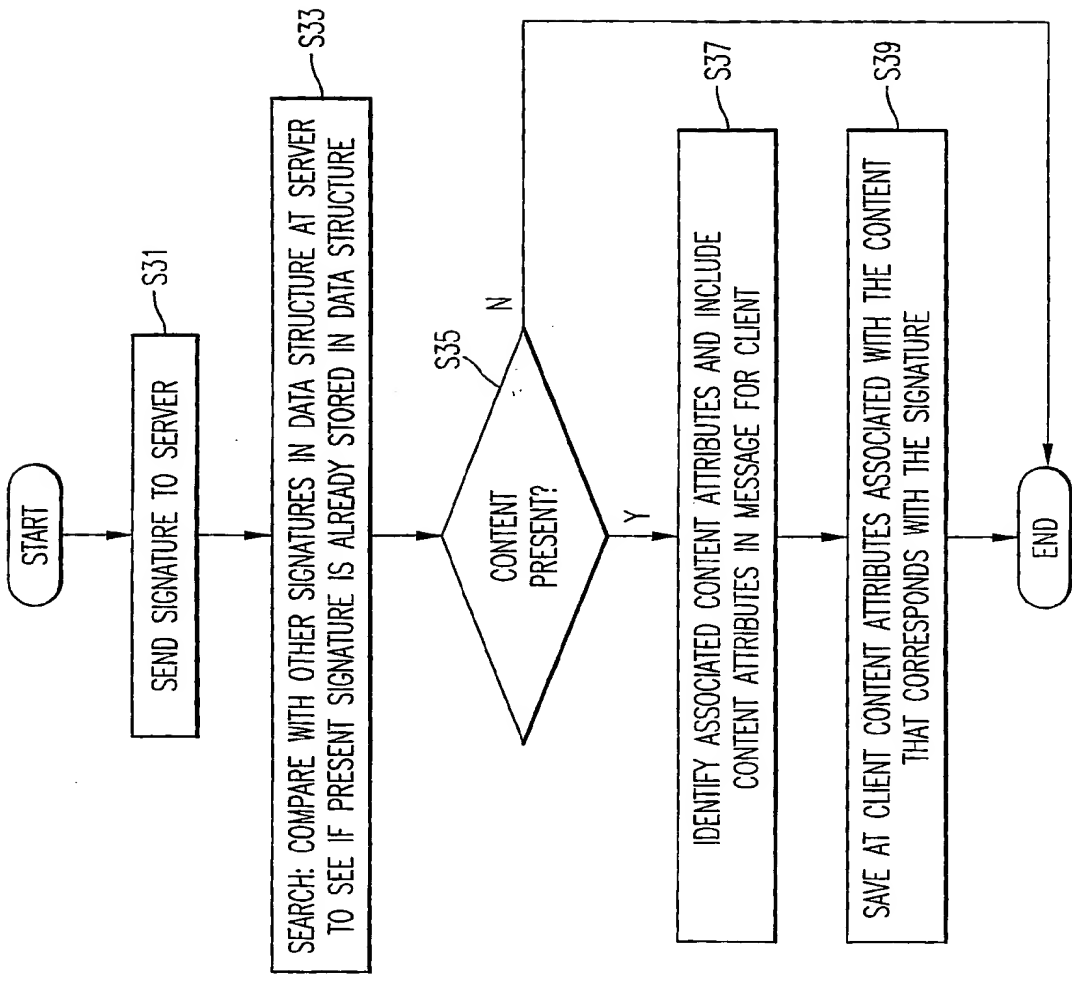

START
IDENTIFY ENCODED CONTENT
S11
PASS CONTENT THROUGH APPROPRIATE CODEC, OBTAIN TIME DOMAIN SIGNAL
S13
FFT TIME DOMAIN SIGNAL AND OBTAIN X SPECTRAL POINTS PER FFT OPERATION
S15
DIVIDE X SPECTRAL POINTS INTO N FREQ. BANDS, EACH WITH A PREDETERMINED WIDTH
S17
AVG. SPECTRAL POINTS FOR EACH BAND OVER ENTIRE CONTENT (S_I)
S19
OBTAIN STD. DEV. FOR AVERAGED SPECTRAL POINTS IN EACH BAND (S_D)
S21
SAVE SIGNATURE AS $S_i(1,x), S_d(1,x), S_i(2,x), S_d(2,x), \ldots S_i(n,x), S_d(n,x)$ IN MEMORY
S23
END

FIG. 5

FIG. 6


START
S31
SEND SIGNATURE TO SERVER
S33
SEARCH: COMPARE WITH OTHER SIGNATURES IN DATA STRUCTURE AT SERVER TO SEE IF PRESENT SIGNATURE IS ALREADY STORED IN DATA STRUCTURE
S35
CONTENT PRESENT?
N
Y
S37
IDENTIFY ASSOCIATED CONTENT ATTRIBUTES AND INCLUDE CONTENT ATTRIBUTES IN MESSAGE FOR CLIENT
S39
SAVE AT CLIENT CONTENT ATTRIBUTES ASSOCIATED WITH THE CONTENT THAT CORRESPONDS WITH THE SIGNATURE
END

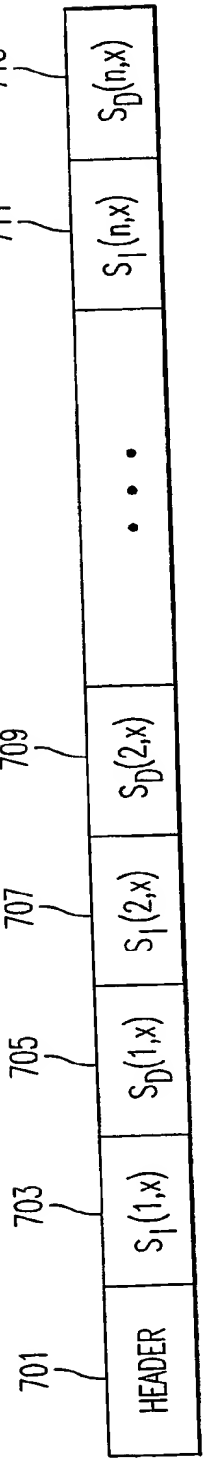

701
HEADER
703
$S_I(1,x)$
705
$S_D(1,x)$
707
$S_I(2,x)$
709
$S_D(2,x)$
711
$S_I(n,x)$
713
$S_D(n,x)$

FIG. 7

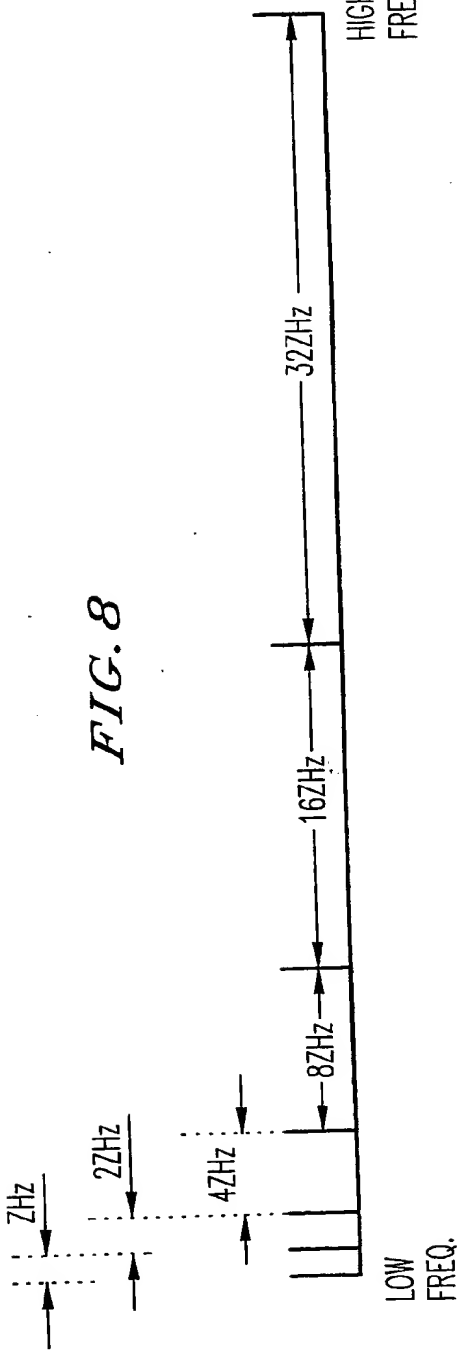

ZHz
2ZHz
4ZHz
8ZHz
16ZHz
32ZHz
LOW FREQ.
HIGH FREQ.

FIG. 8

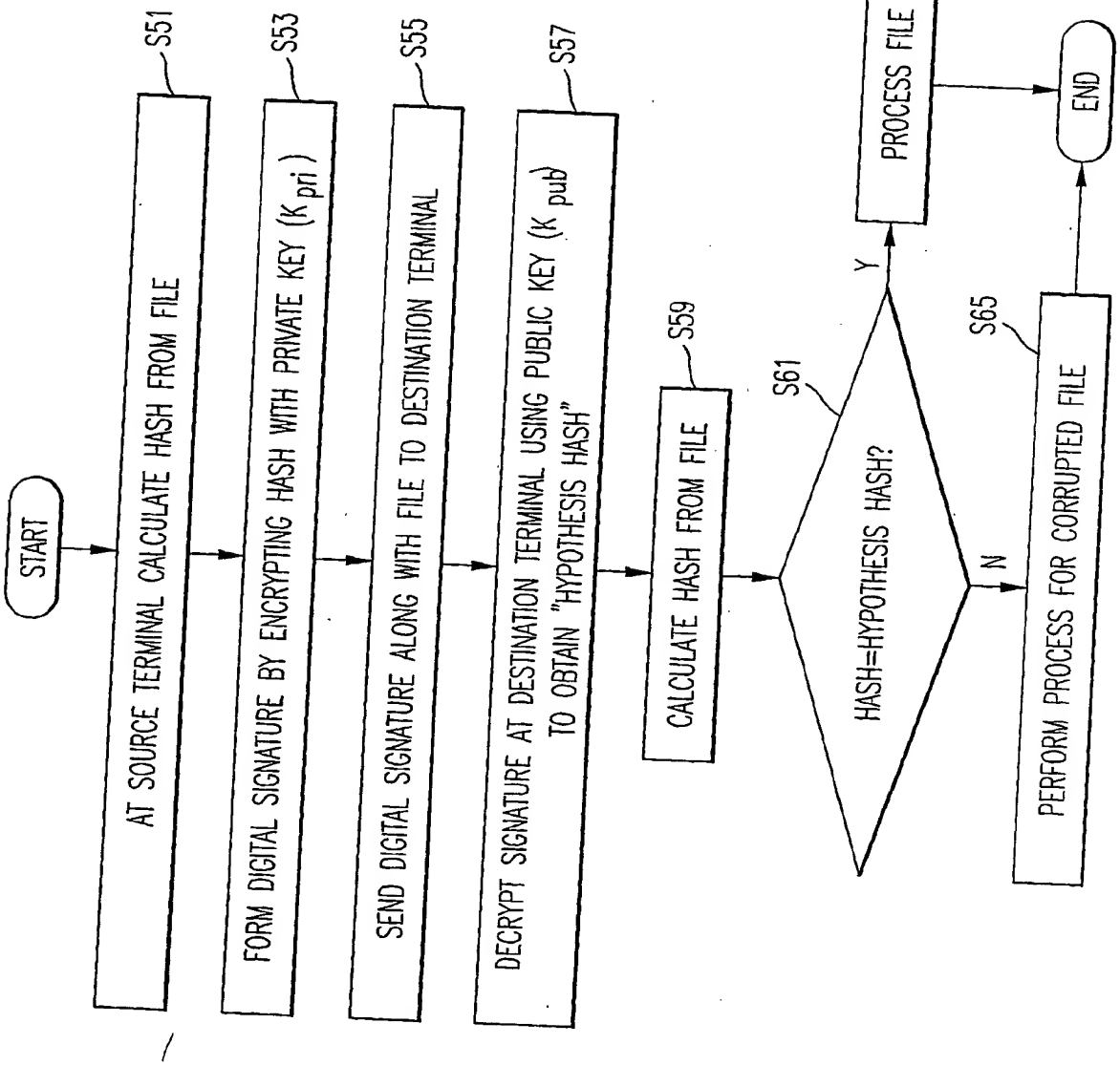


FIG. 9
(BACKGROUND ART)

FIG. 10

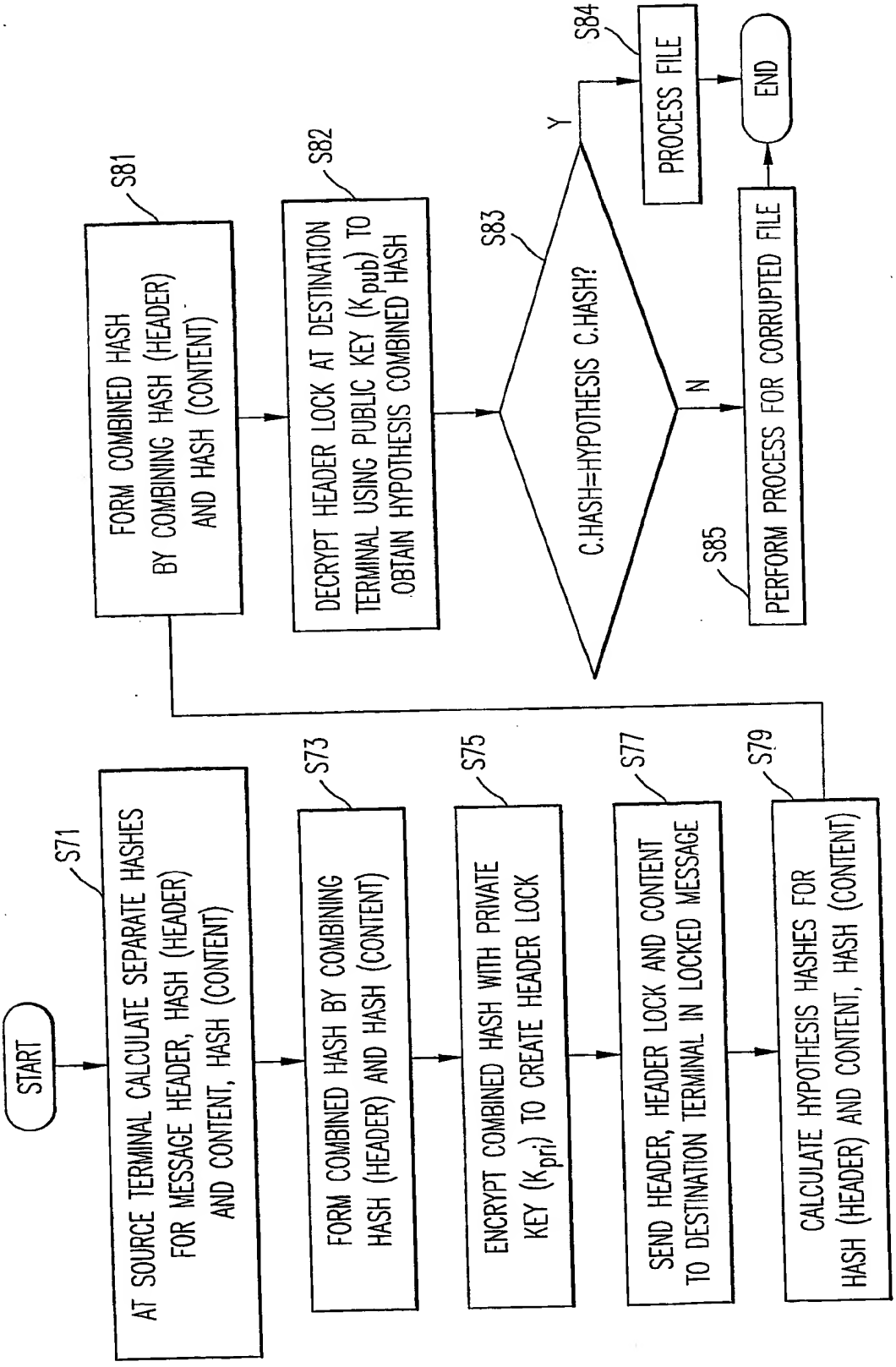

START
S71
AT SOURCE TERMINAL CALCULATE SEPARATE HASHES FOR MESSAGE HEADER, HASH (HEADER) AND CONTENT, HASH (CONTENT)
S73
FORM COMBINED HASH BY COMBINING HASH (HEADER) AND HASH (CONTENT)
S75
ENCRYPT COMBINED HASH WITH PRIVATE KEY (K_{pri}) TO CREATE HEADER LOCK
S77
SEND HEADER, HEADER LOCK AND CONTENT TO DESTINATION TERMINAL IN LOCKED MESSAGE
S79
CALCULATE HYPOTHESIS HASHES FOR HASH (HEADER) AND CONTENT, HASH (CONTENT)
S81
FORM COMBINED HASH BY COMBINING HASH (HEADER) AND HASH (CONTENT)
S82
DECRYPT HEADER LOCK AT DESTINATION TERMINAL USING PUBLIC KEY (K_{pub}) TO OBTAIN HYPOTHESIS COMBINED HASH
S83
C.HASH=HYPOTHESIS C.HASH?
Y
N
S84
PROCESS FILE
S85
PERFORM PROCESS FOR CORRUPTED FILE
END

FIG. 11

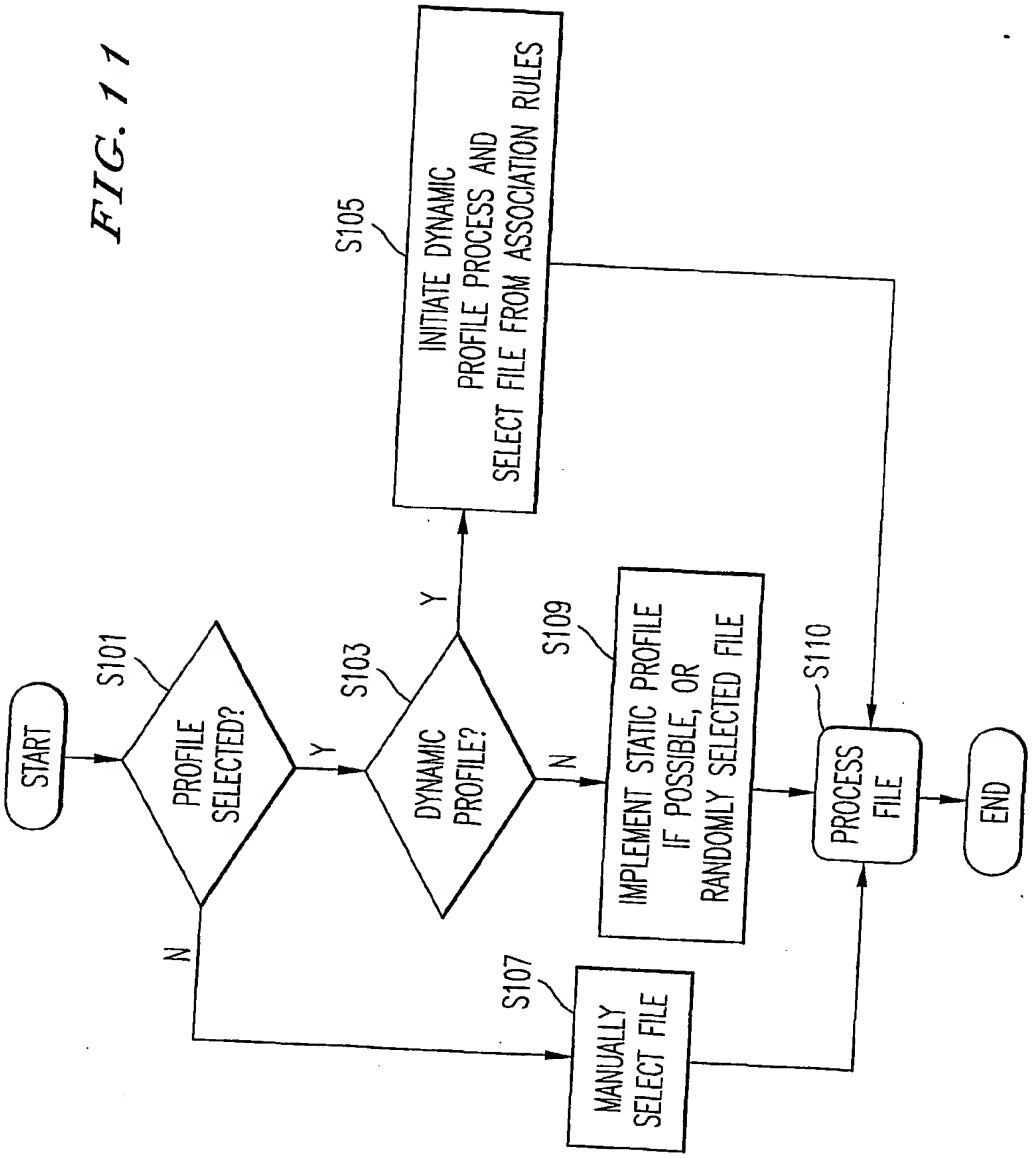

START
S101
PROFILE SELECTED?
Y
S103
DYNAMIC PROFILE?
Y
S105
INITIATE DYNAMIC PROFILE PROCESS AND SELECT FILE FROM ASSOCIATION RULES
N
S109
IMPLEMENT STATIC PROFILE IF POSSIBLE, OR RANDOMLY SELECTED FILE
N
S107
MANUALLY SELECT FILE
S110
PROCESS FILE
END

FIG. 12

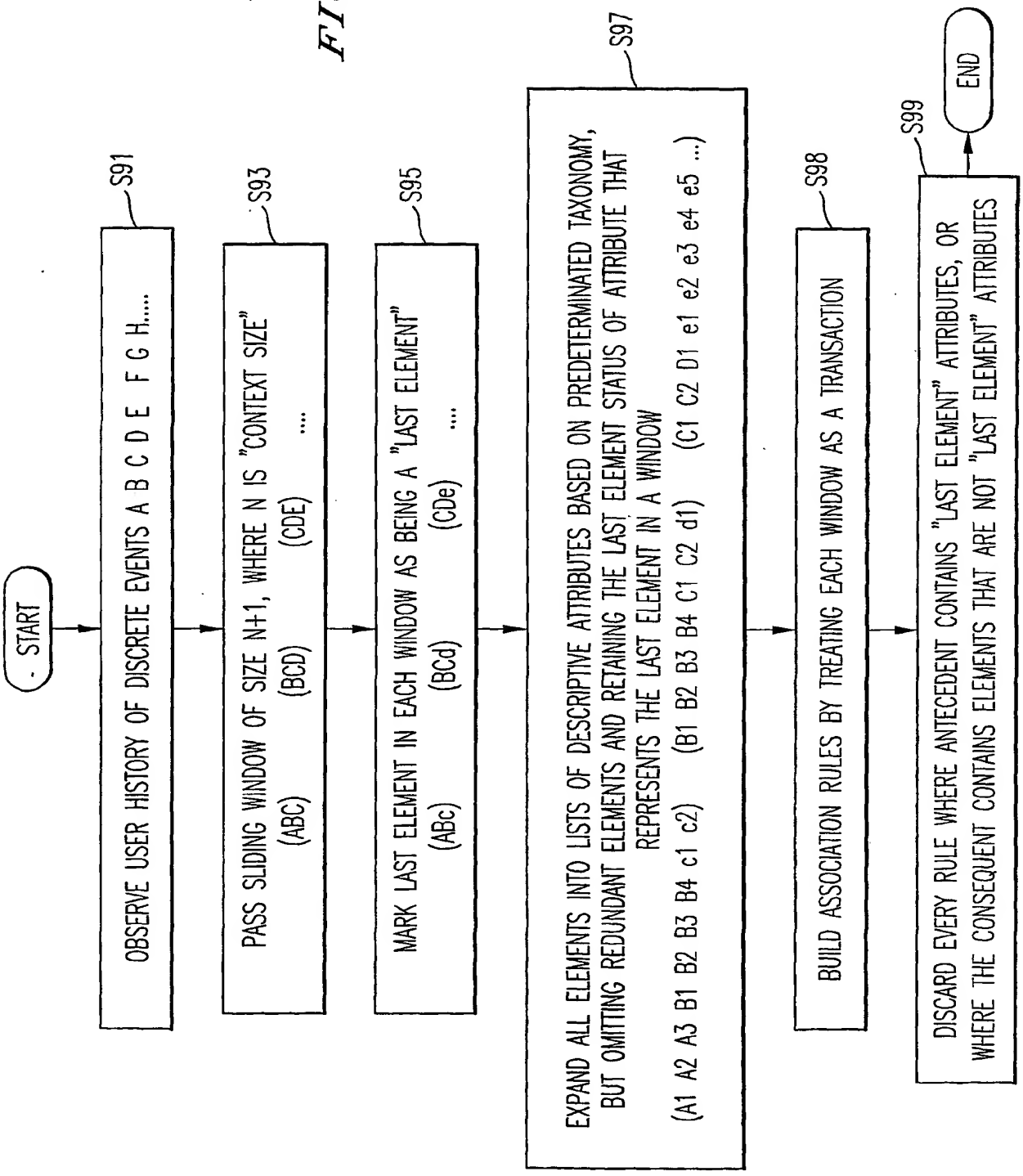

START
S91
OBSERVE USER HISTORY OF DISCRETE EVENTS A B C D E F G H.....
S93
PASS SLIDING WINDOW OF SIZE N+1, WHERE N IS "CONTEXT SIZE"
(ABC) (BCD) (CDE) ....
S95
MARK LAST ELEMENT IN EACH WINDOW AS BEING A "LAST ELEMENT"
(ABc) (BCd) (CDe) ....
S97
EXPAND ALL ELEMENTS INTO LISTS OF DESCRIPTIVE ATTRIBUTES BASED ON PREDETERMINATED TAXONOMY, BUT OMITTING REDUNDANT ELEMENTS AND RETAINING THE LAST ELEMENT STATUS OF ATTRIBUTE THAT REPRESENTS THE LAST ELEMENT IN A WINDOW
(A1 A2 A3 B1 B2 B3 B4 c1 c2) (B1 B2 B3 B4 C1 C2 d1) (C1 C2 D1 e1 e2 e3 e4 e5 ...)
S98
BUILD ASSOCIATION RULES BY TREATING EACH WINDOW AS A TRANSACTION
S99
DISCARD EVERY RULE WHERE ANTECEDENT CONTAINS "LAST ELEMENT" ATTRIBUTES, OR WHERE THE CONSEQUENT CONTAINS ELEMENTS THAT ARE NOT "LAST ELEMENT" ATTRIBUTES
END

FIG. 13

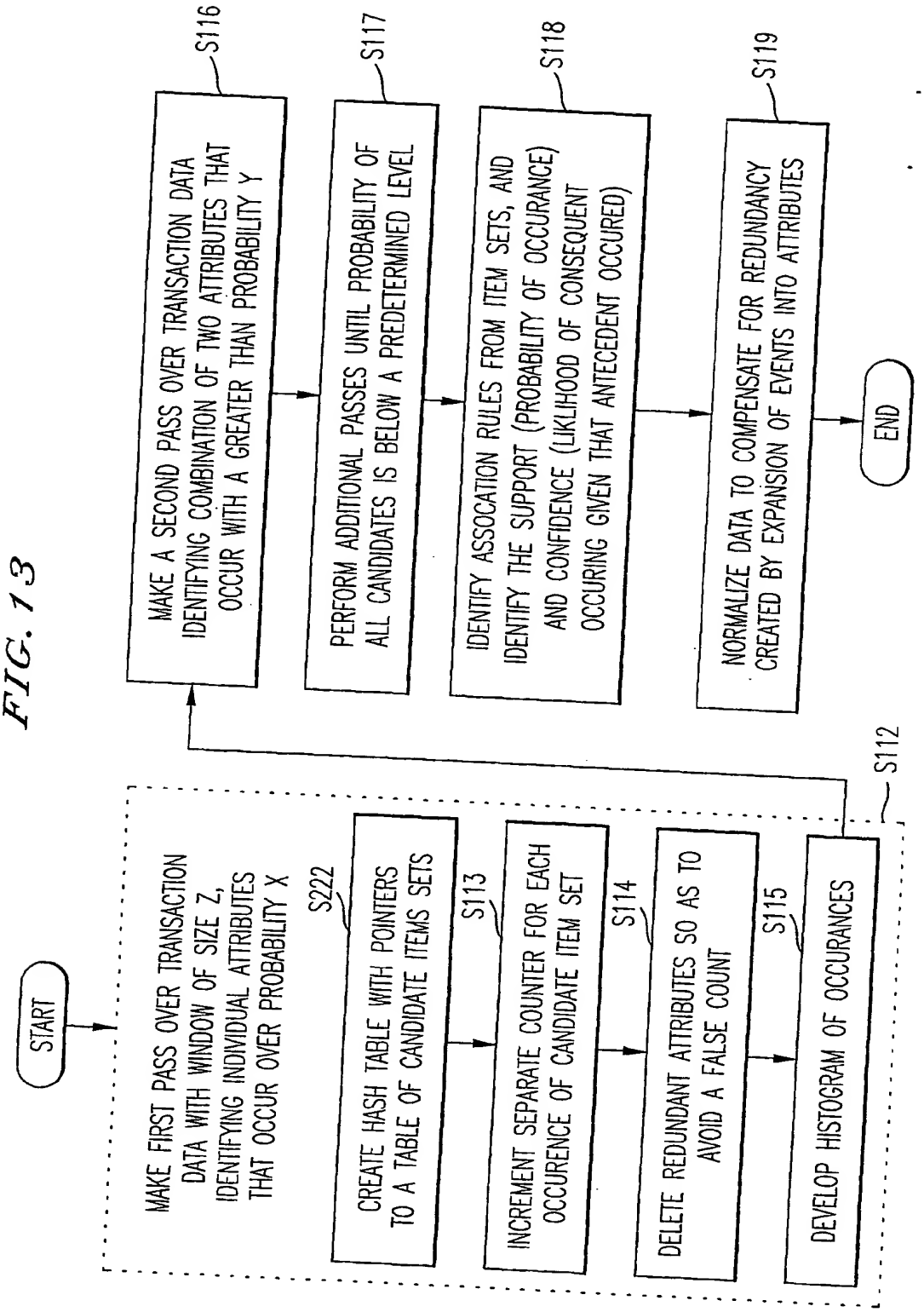

START
MAKE FIRST PASS OVER TRANSACTION DATA WITH WINDOW OF SIZE Z, IDENTIFYING INDIVIDUAL ATTRIBUTES THAT OCCUR OVER PROBABILITY X
S222
CREATE HASH TABLE WITH POINTERS TO A TABLE OF CANDIDATE ITEMS SETS
S113
INCREMENT SEPARATE COUNTER FOR EACH OCCURENCE OF CANDIDATE ITEM SET
S114
DELETE REDUNDANT ATTRIBUTES SO AS TO AVOID A FALSE COUNT
S115
DEVELOP HISTOGRAM OF OCCURANCES
S112
MAKE A SECOND PASS OVER TRANSACTION DATA IDENTIFYING COMBINATION OF TWO ATTRIBUTES THAT OCCUR WITH A GREATER THAN PROBABILITY Y
S116
PERFORM ADDITIONAL PASSES UNTIL PROBABILITY OF ALL CANDIDATES IS BELOW A PREDETERMINED LEVEL
S117
IDENTIFY ASSOCATION RULES FROM ITEM SETS, AND IDENTIFY THE SUPPORT (PROBABILITY OF OCCURANCE) AND CONFIDENCE (LIKLIHOOD OF CONSEQUENT OCCURING GIVEN THAT ANTECEDENT OCCURED)
S118
NORMALIZE DATA TO COMPENSATE FOR REDUNDANCY CREATED BY EXPANSION OF EVENTS INTO ATTRIBUTES
S119
END

FIG. 14

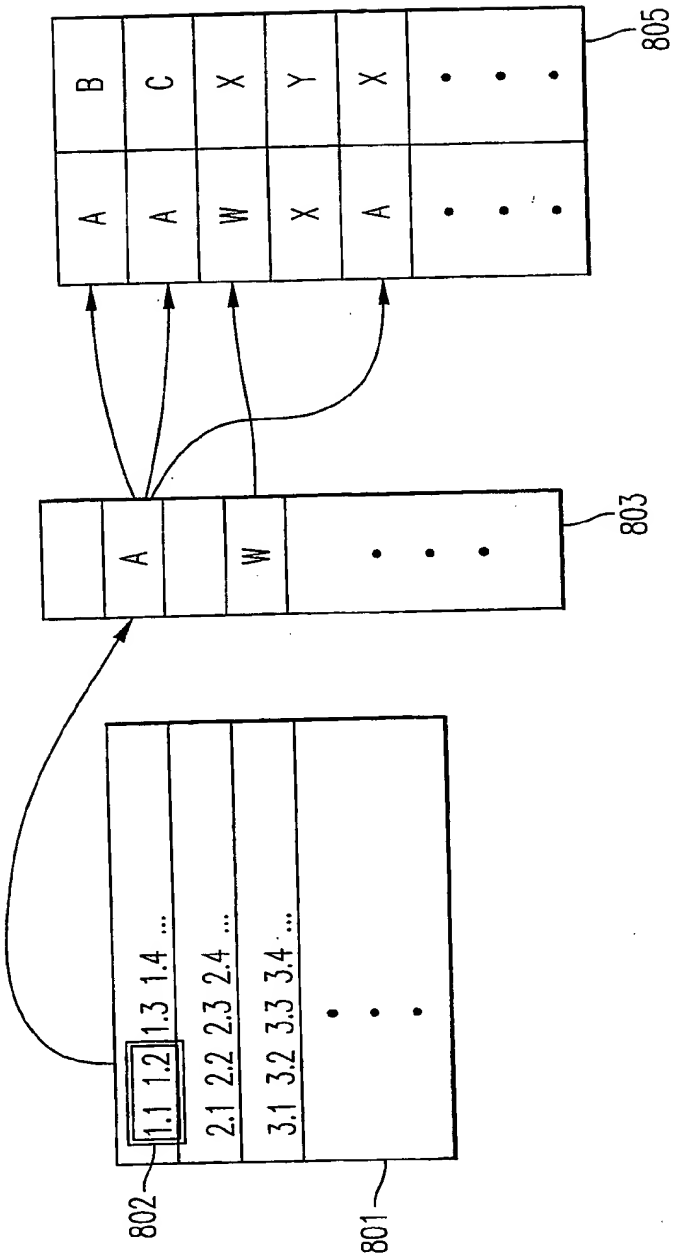


FIG. 15a

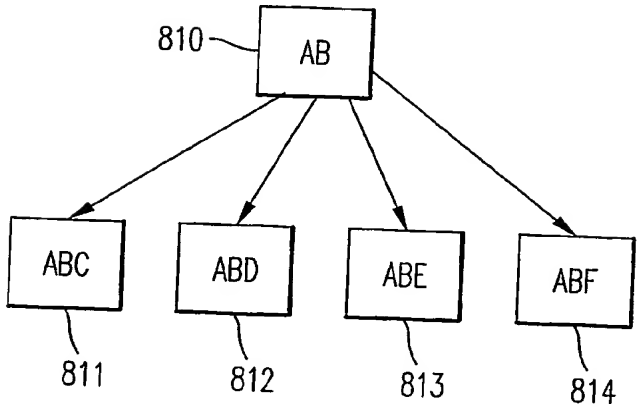


FIG. 15b

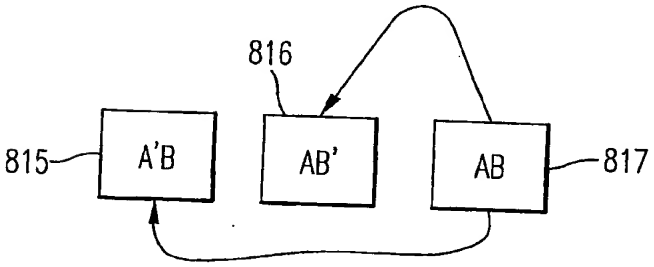


FIG. 15c

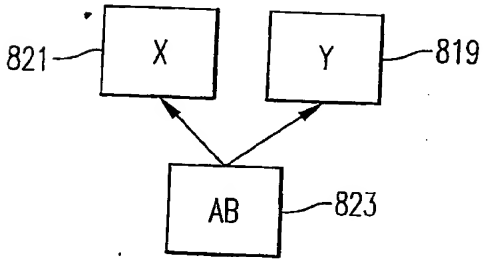
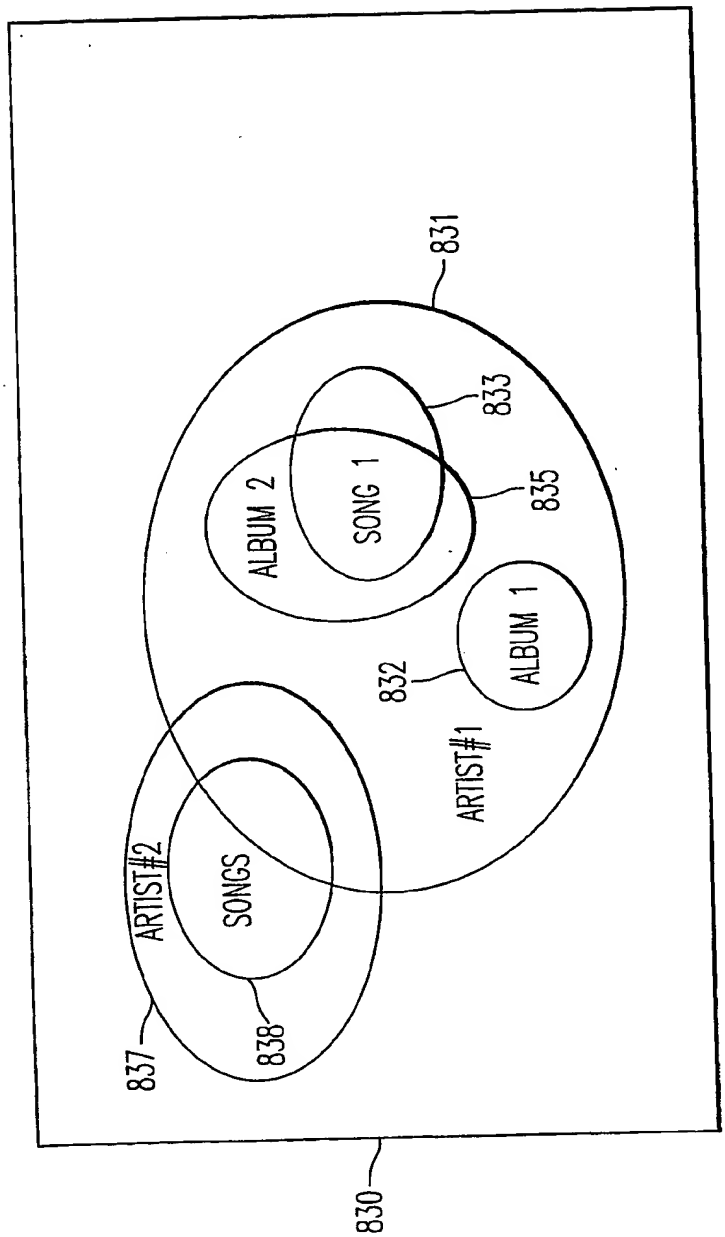


FIG. 16


830
831
833
835
832
837
838
ARTIST#1
ALBUM 1
ALBUM 2
SONG 1
ARTIST#2
SONGS

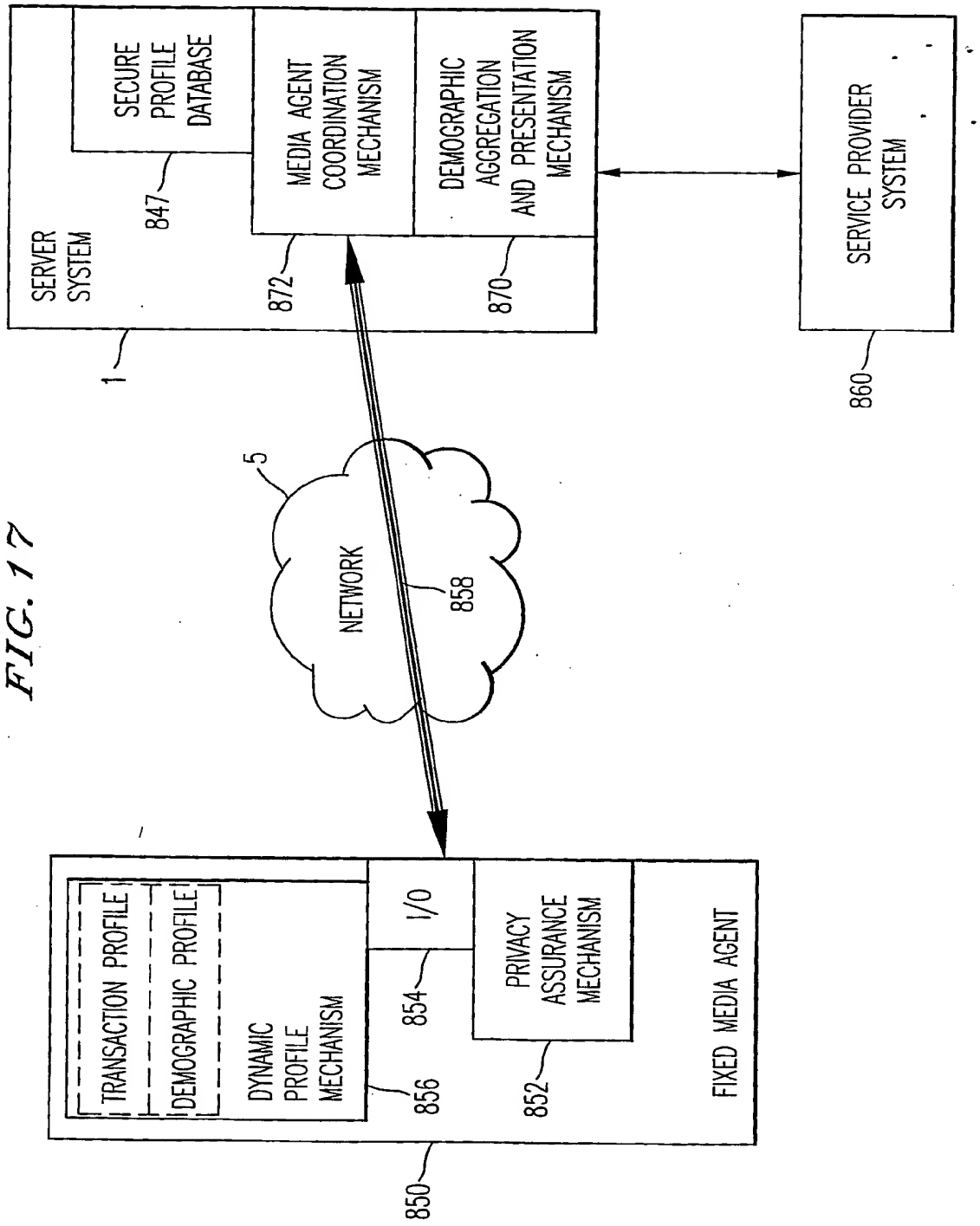

SERVER SYSTEM
1
SECURE PROFILE DATABASE
847
MEDIA AGENT COORDINATION MECHANISM
872
DEMOGRAPHIC AGGREGATION AND PRESENTATION MECHANISM
870
SERVICE PROVIDER SYSTEM
860
5
NETWORK
858
TRANSACTION PROFILE
DEMOGRAPHIC PROFILE
DYNAMIC PROFILE MECHANISM
856
I/O
854
PRIVACY ASSURANCE MECHANISM
852
FIXED MEDIA AGENT
850

FIG. 17

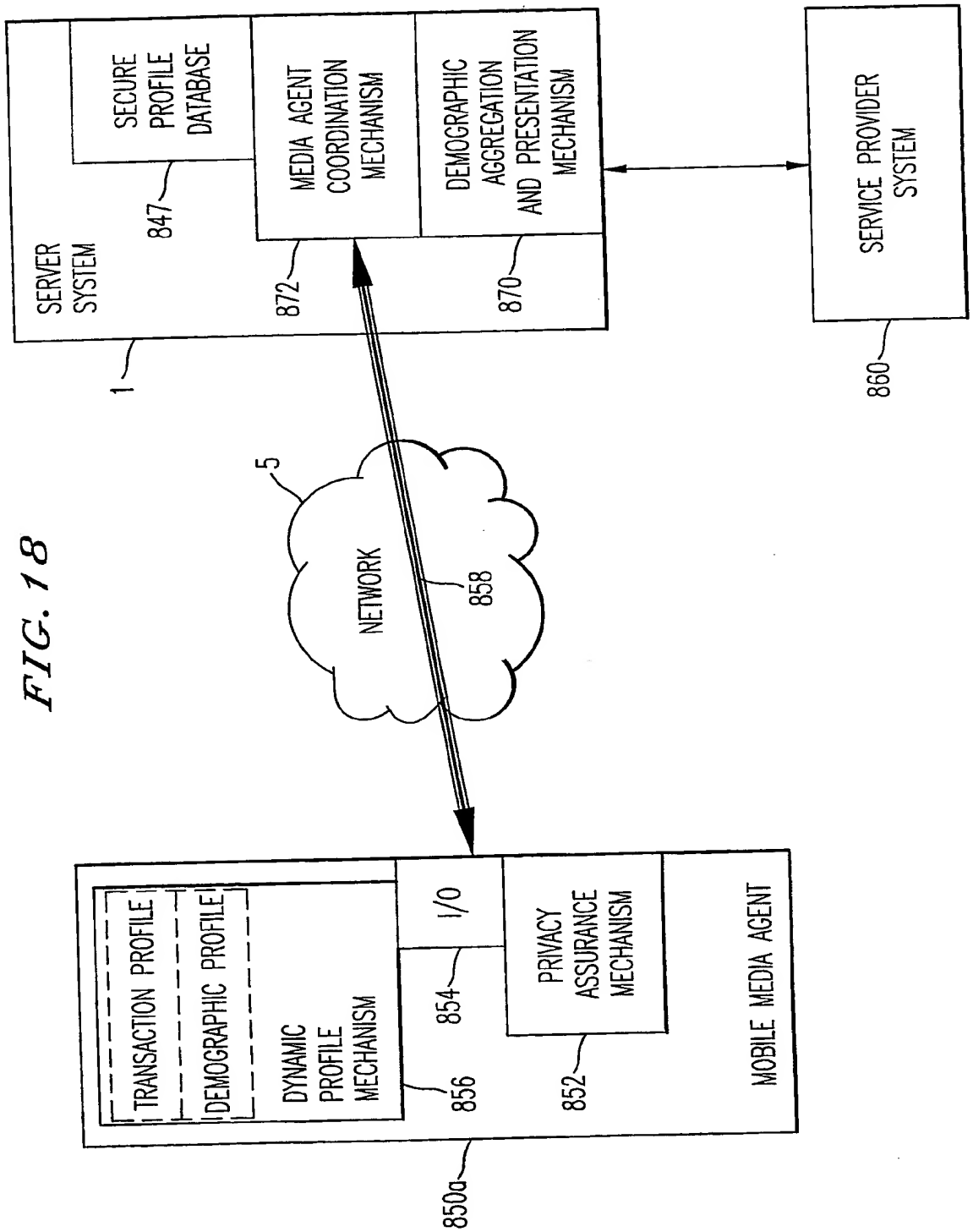

SERVER SYSTEM
1
SECURE PROFILE DATABASE
847
MEDIA AGENT COORDINATION MECHANISM
872
DEMOGRAPHIC AGGREGATION AND PRESENTATION MECHANISM
870
SERVICE PROVIDER SYSTEM
860
5
NETWORK
858
TRANSACTION PROFILE
DEMOGRAPHIC PROFILE
DYNAMIC PROFILE MECHANISM
856
I/O
854
PRIVACY ASSURANCE MECHANISM
852
MOBILE MEDIA AGENT
850a

FIG. 18

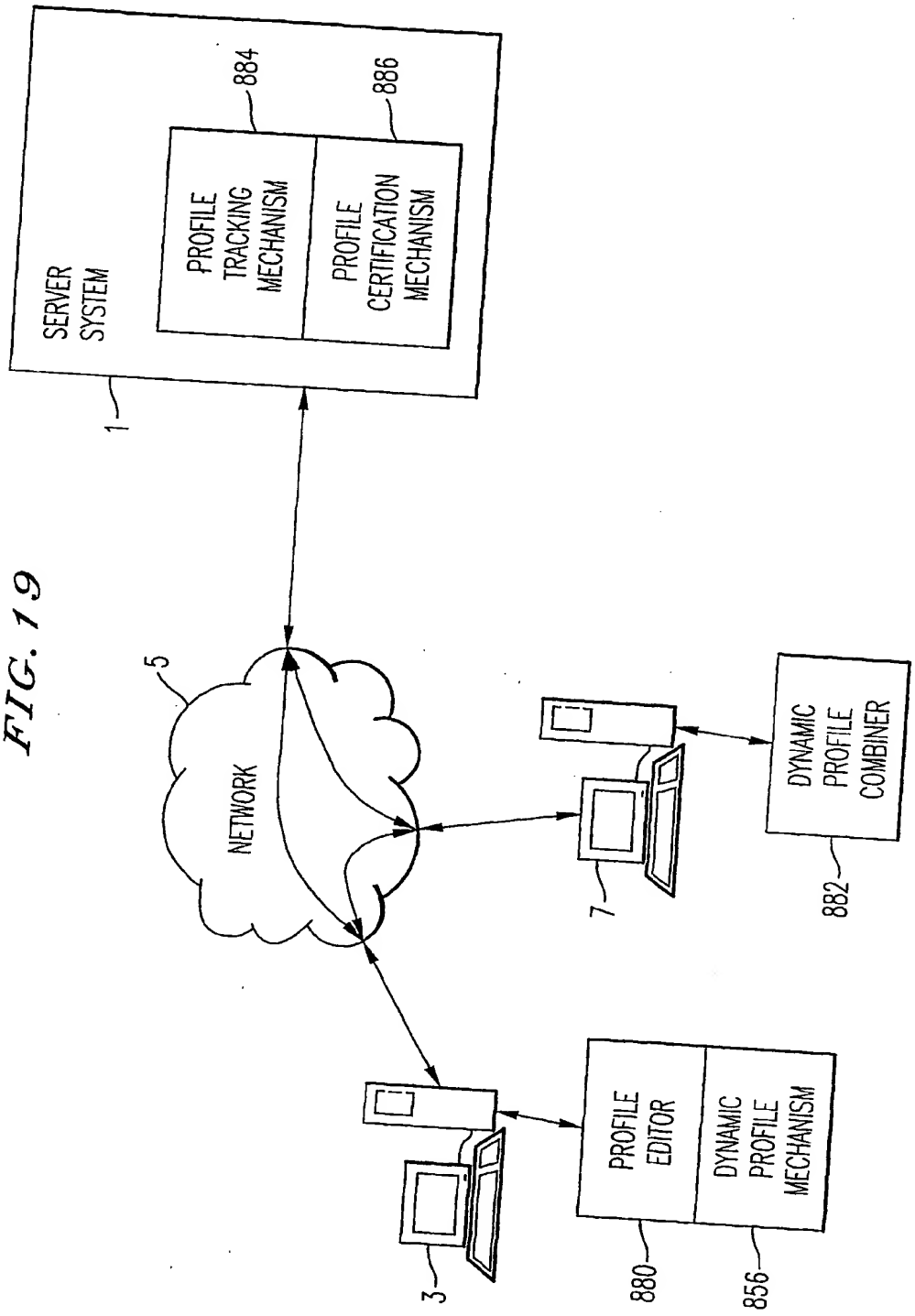

SERVER SYSTEM
1
884
886
PROFILE TRACKING MECHANISM
PROFILE CERTIFICATION MECHANISM
5
NETWORK
7
DYNAMIC PROFILE COMBINER
882
3
PROFILE EDITOR
DYNAMIC PROFILE MECHANISM
880
856

FIG. 19

FIG. 20

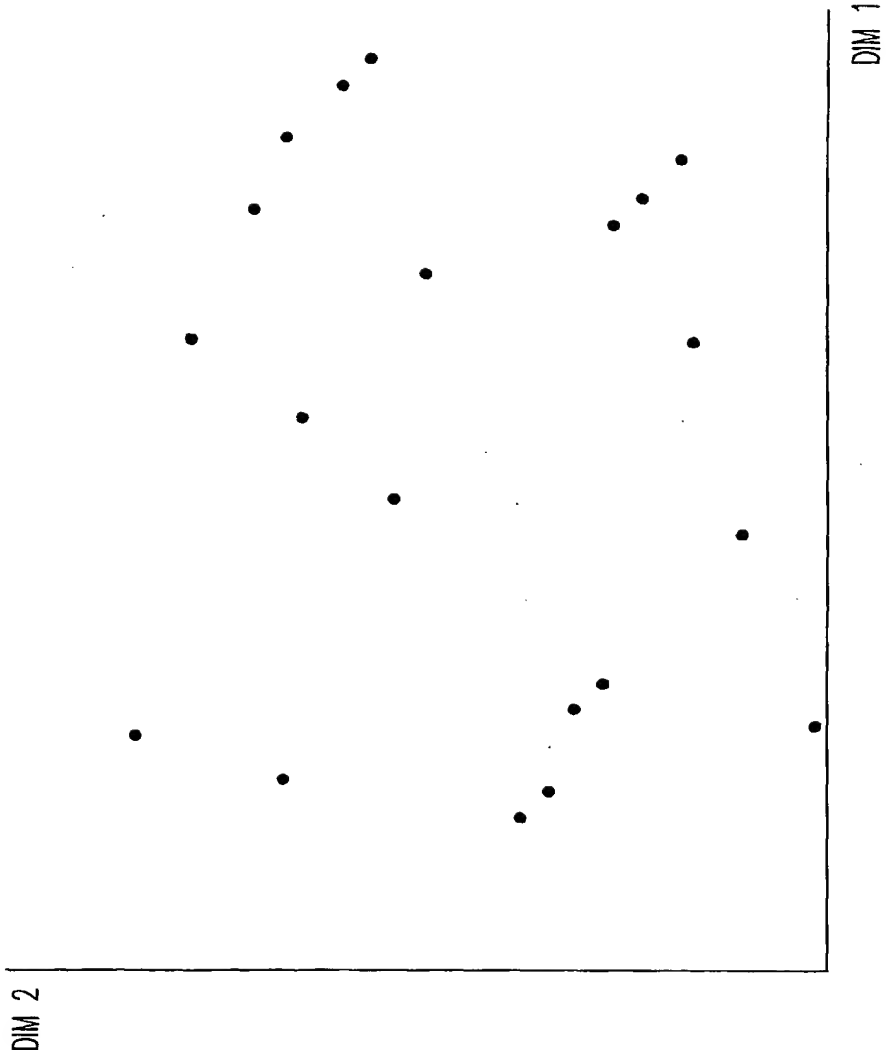


FIG. 21

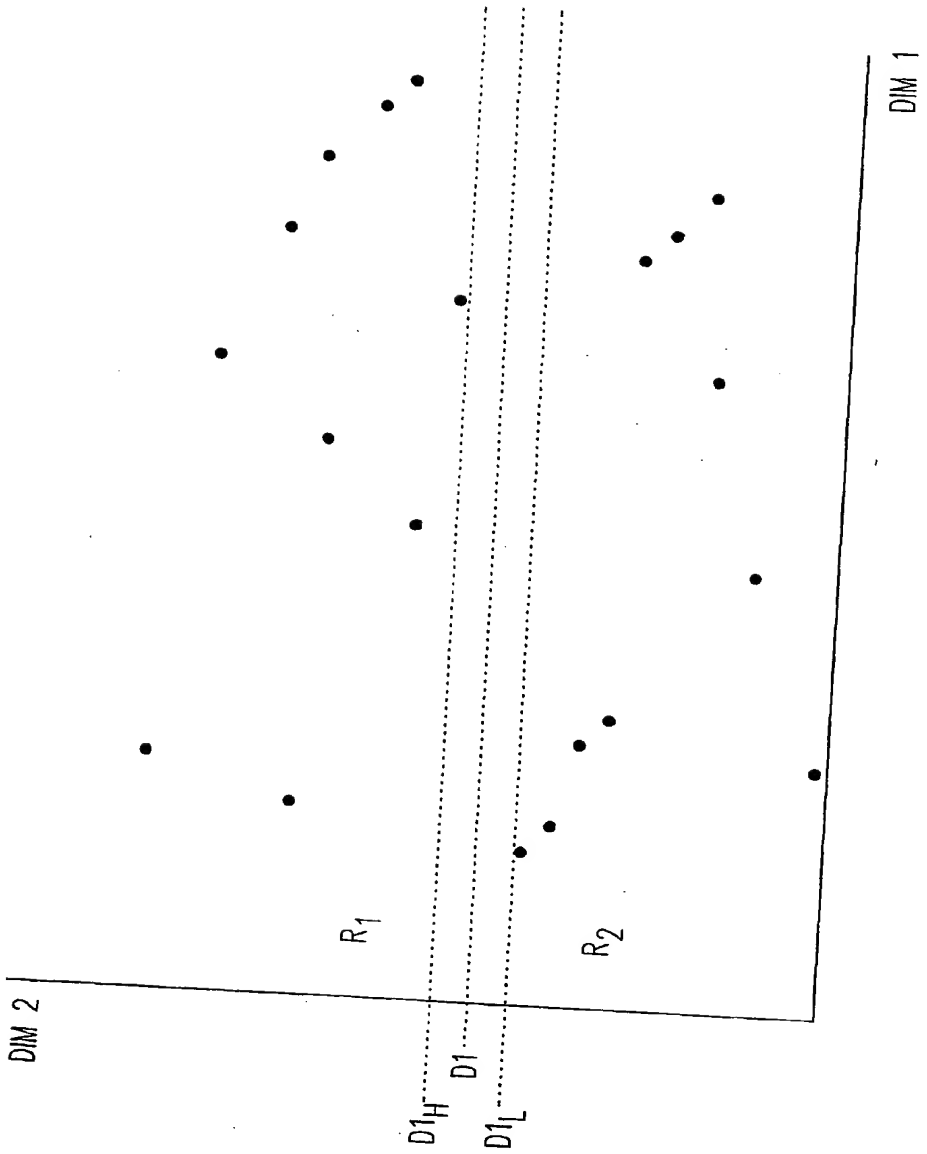
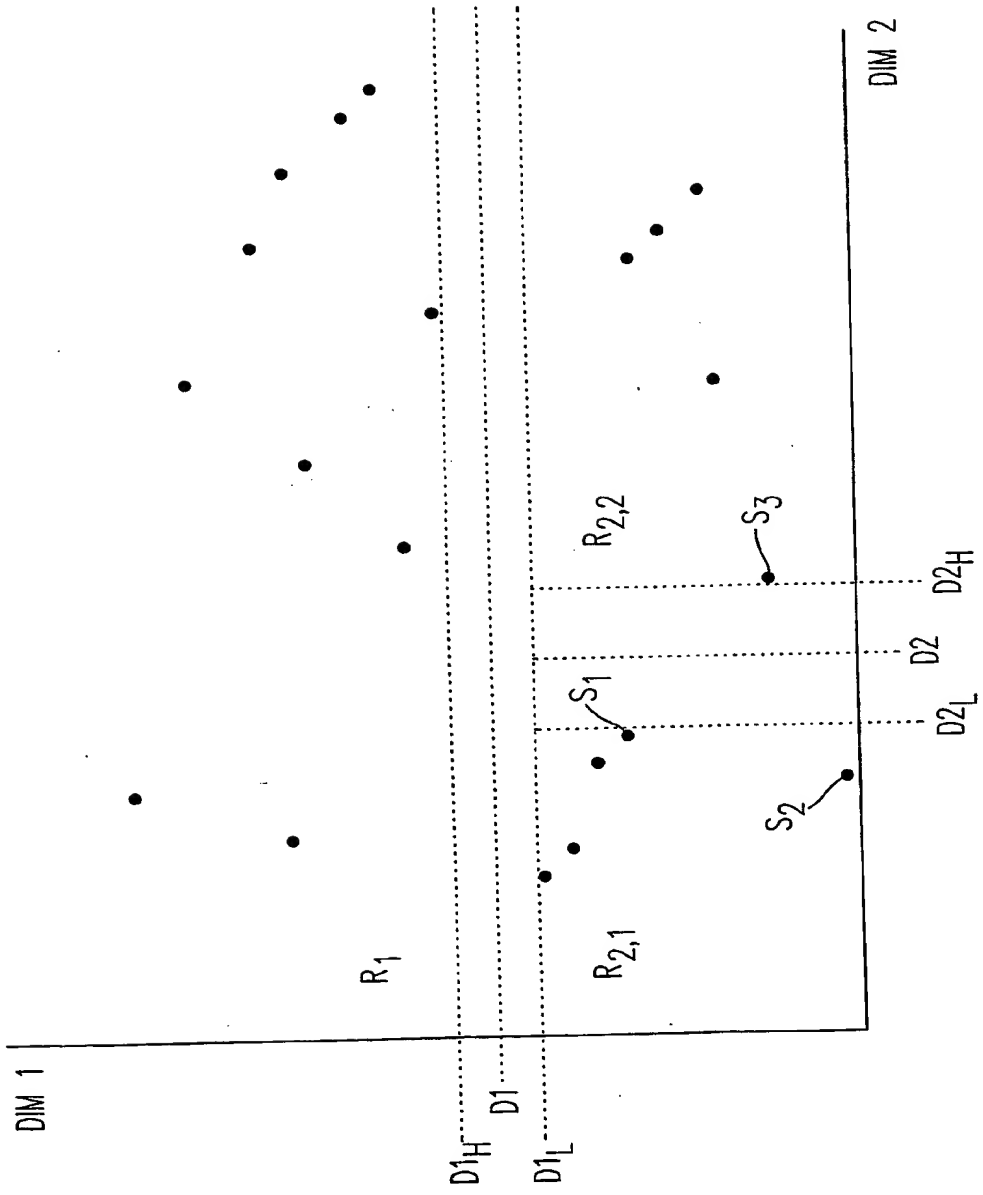


FIG. 22



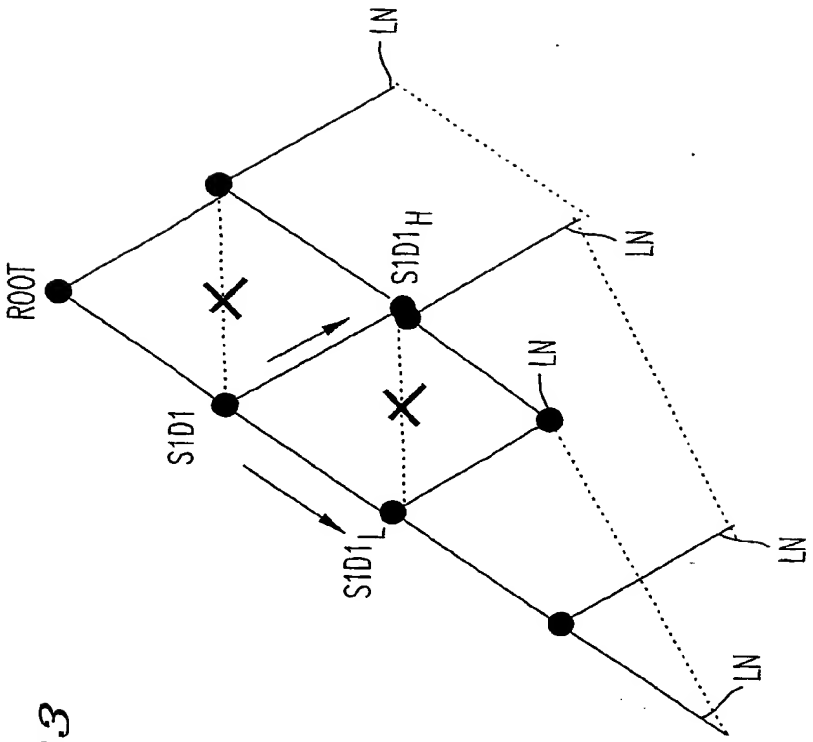
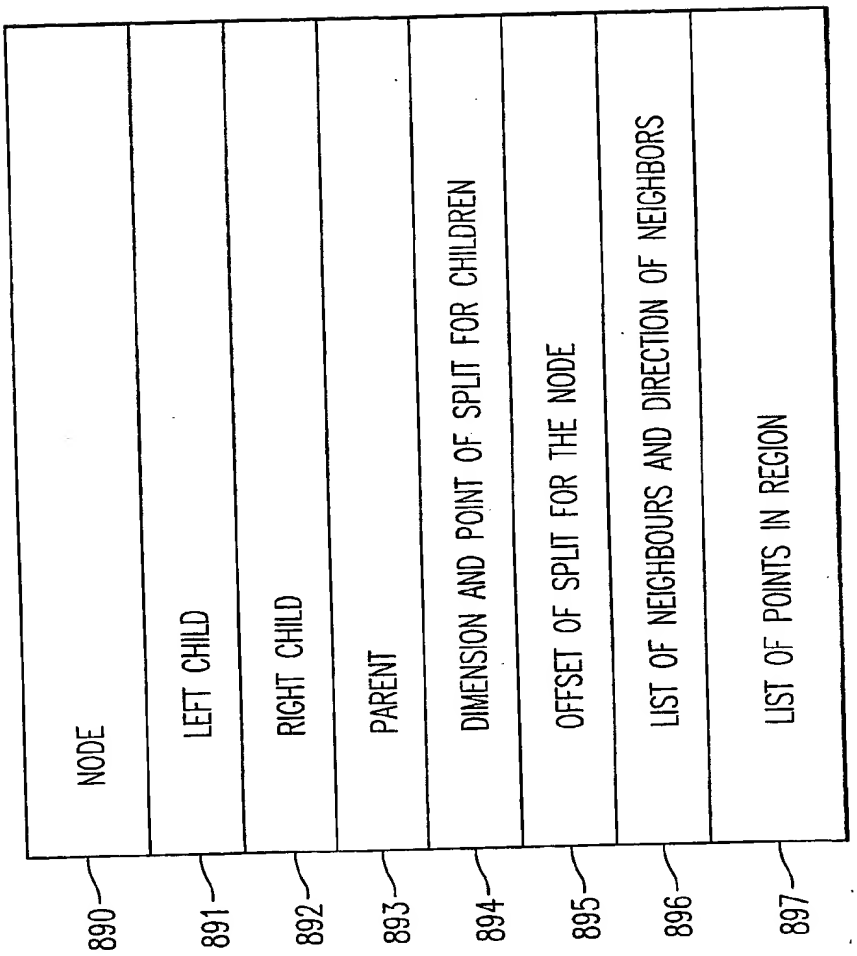


FIG. 23

FIG. 24


NODE
LEFT CHILD
RIGHT CHILD
PARENT
DIMENSION AND POINT OF SPLIT FOR CHILDREN
OFFSET OF SPLIT FOR THE NODE
LIST OF NEIGHBOURS AND DIRECTION OF NEIGHBORS
LIST OF POINTS IN REGION
890
891
892
893
894
895
896
897

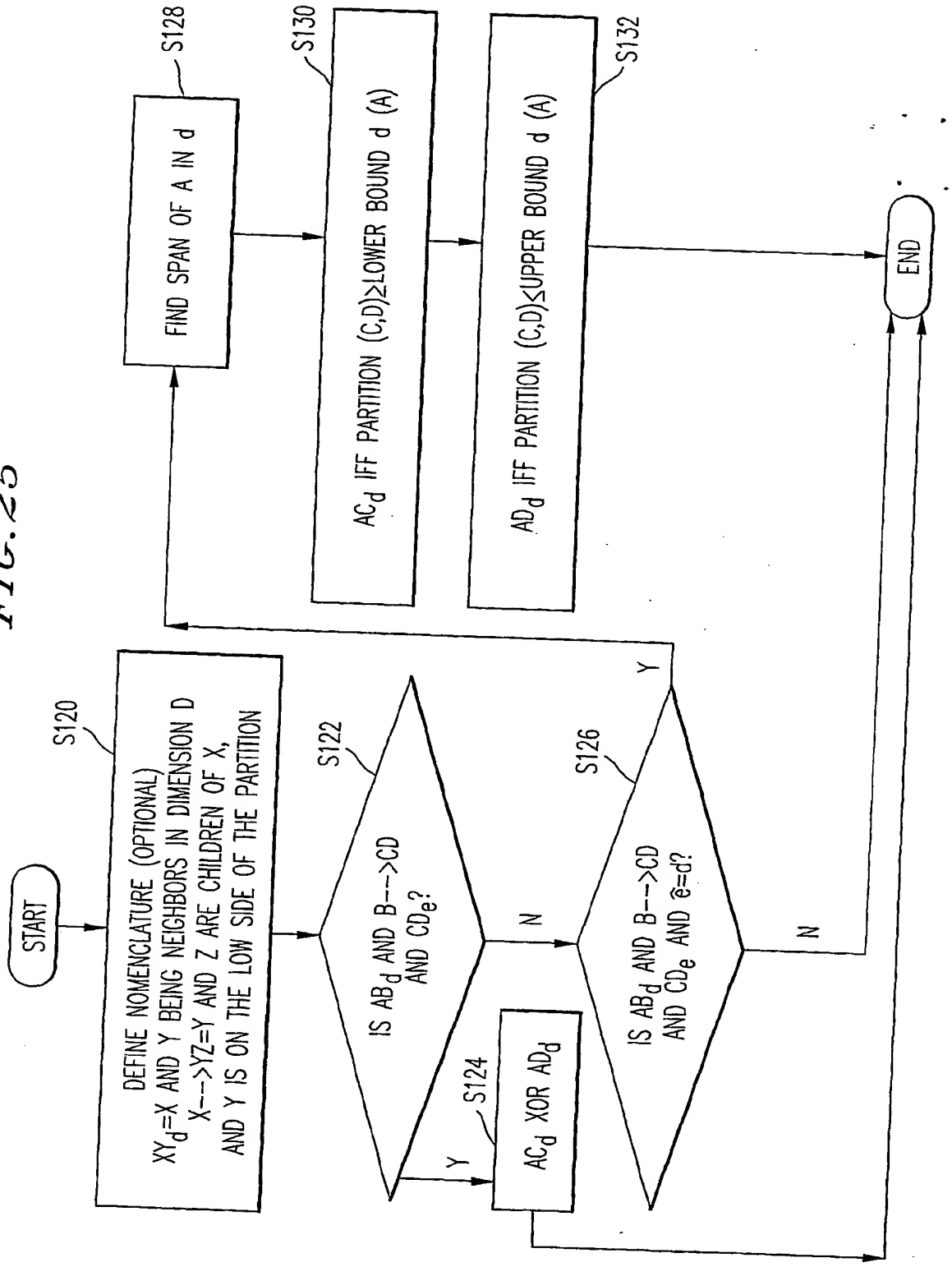

START
S120
DEFINE NOMENCLATURE (OPTIONAL)
XY_d=X AND Y BEING NEIGHBORS IN DIMENSION D
X-->YZ=Y AND Z ARE CHILDREN OF X,
AND Y IS ON THE LOW SIDE OF THE PARTITION
S122
IS AB_d AND B-->CD AND CD_e?
N
Y
S124
AC_d XOR AD_d
S126
IS AB_d AND B-->CD AND CD_e AND e=d?
Y
N
S128
FIND SPAN OF A IN d
S130
AC_d IFF PARTITION (C,D)≥LOWER BOUND d (A)
S132
AD_d IFF PARTITION (C,D)≤UPPER BOUND d (A)
END

FIG. 25

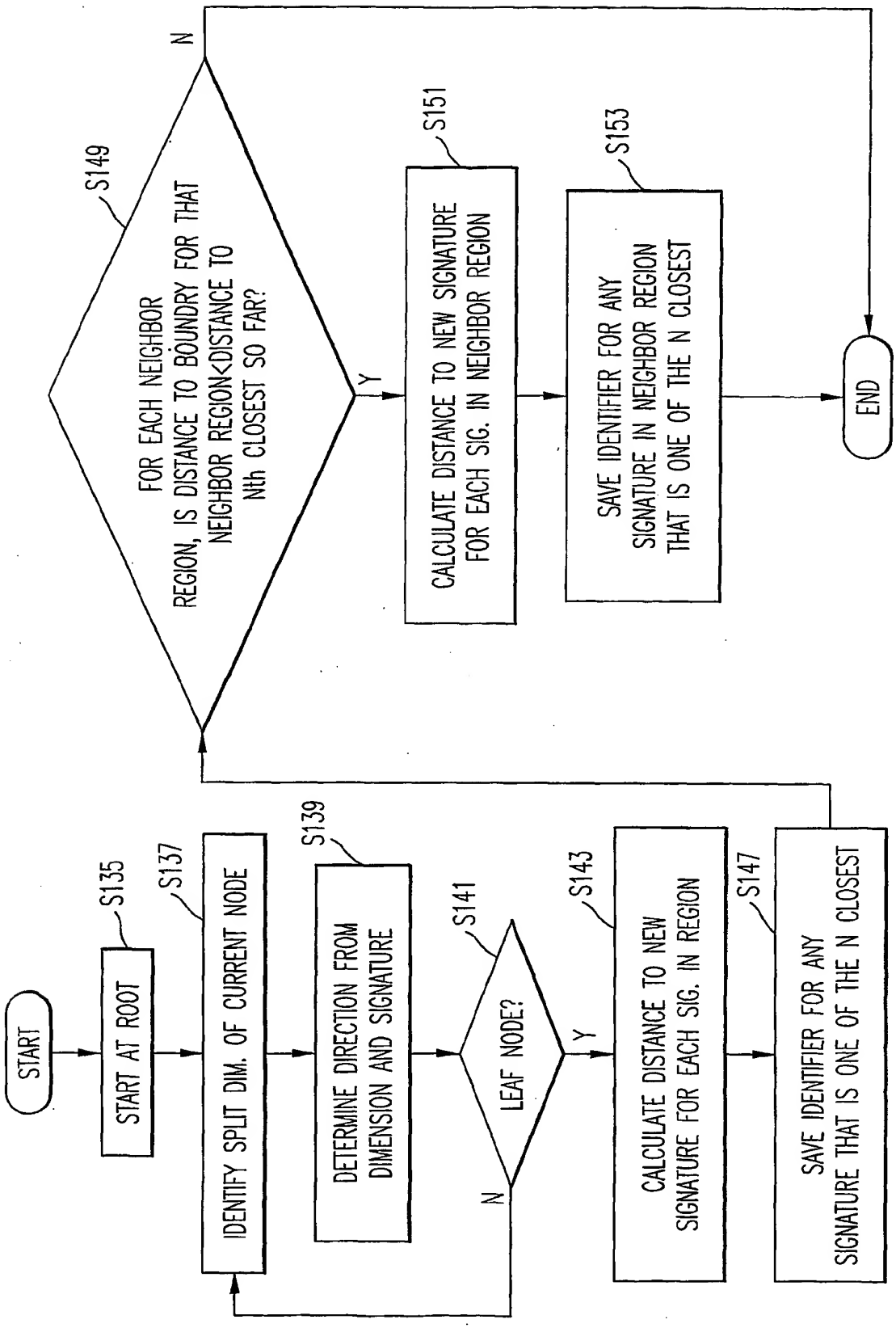
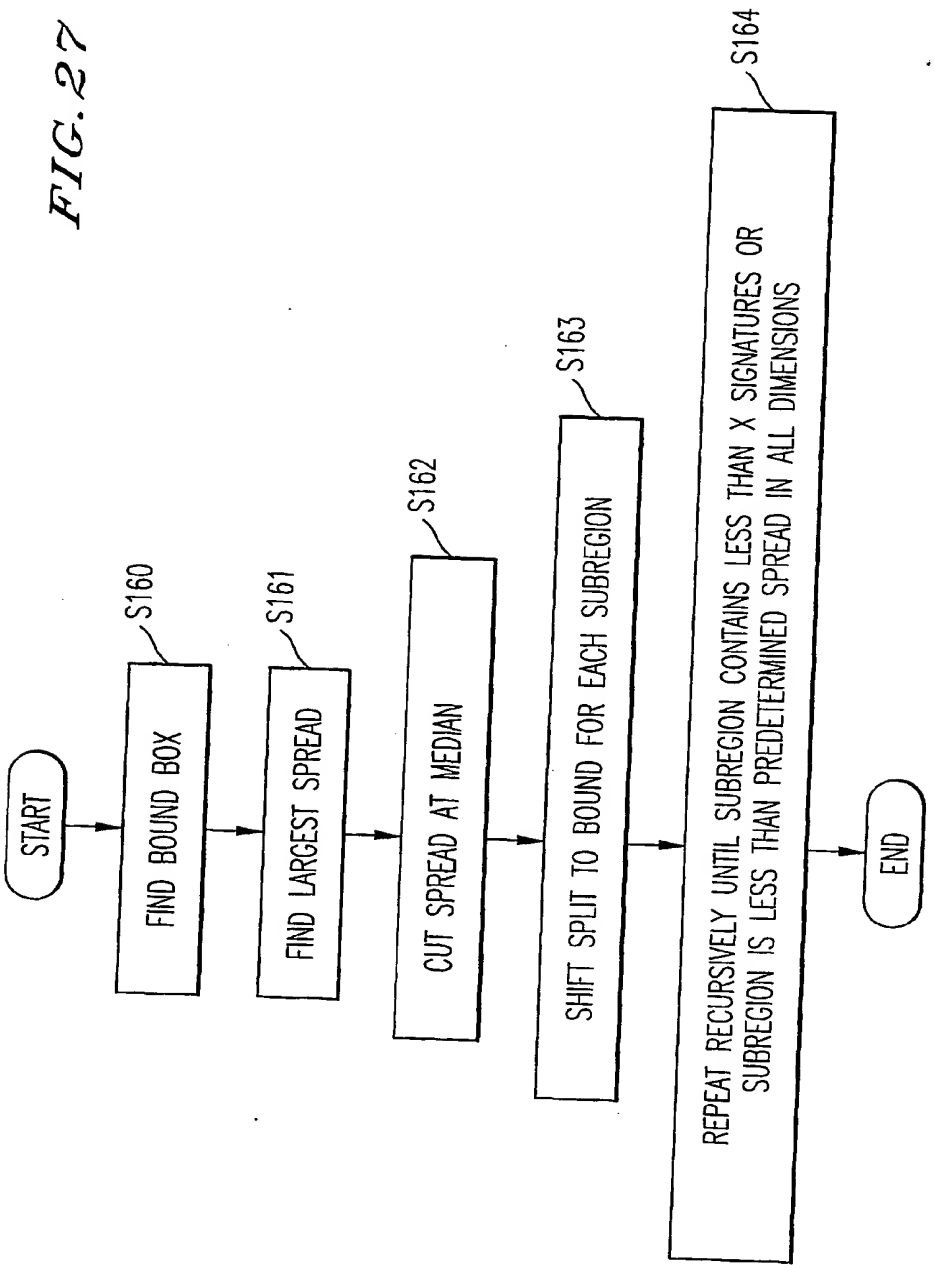

START
START AT ROOT
S135
IDENTIFY SPLIT DIM. OF CURRENT NODE
S137
DETERMINE DIRECTION FROM DIMENSION AND SIGNATURE
S139
LEAF NODE?
S141
N
Y
CALCULATE DISTANCE TO NEW SIGNATURE FOR EACH SIG. IN REGION
S143
SAVE IDENTIFIER FOR ANY SIGNATURE THAT IS ONE OF THE N CLOSEST
S147
FOR EACH NEIGHBOR REGION, IS DISTANCE TO BOUNDRY FOR THAT NEIGHBOR REGION<DISTANCE TO Nth CLOSEST SO FAR?
S149
N
Y
CALCULATE DISTANCE TO NEW SIGNATURE FOR EACH SIG. IN NEIGHBOR REGION
S151
SAVE IDENTIFIER FOR ANY SIGNATURE IN NEIGHBOR REGION THAT IS ONE OF THE N CLOSEST
S153
END

FIG. 26

FIG. 27


START
FIND BOUND BOX
S160
FIND LARGEST SPREAD
S161
CUT SPREAD AT MEDIAN
S162
SHIFT SPLIT TO BOUND FOR EACH SUBREGION
S163
REPEAT RECURSIVELY UNTIL SUBREGION CONTAINS LESS THAN X SIGNATURES OR SUBREGION IS LESS THAN PREDETERMINED SPREAD IN ALL DIMENSIONS
S164
END

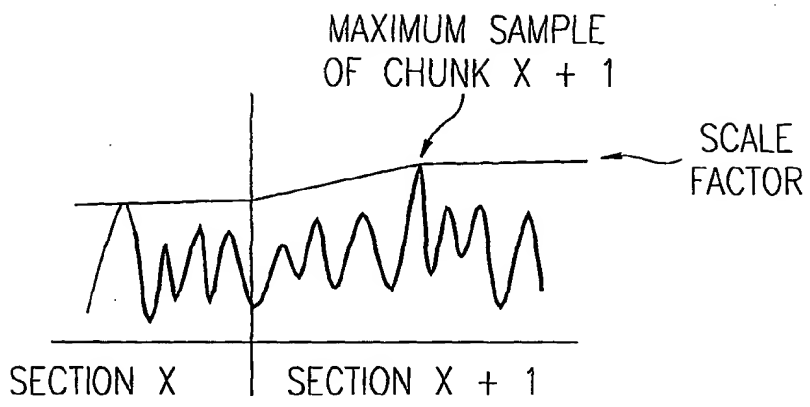


FIG. 28

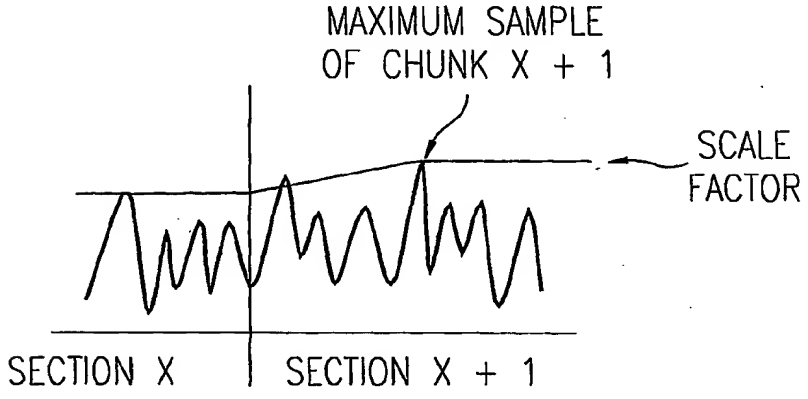


FIG. 29

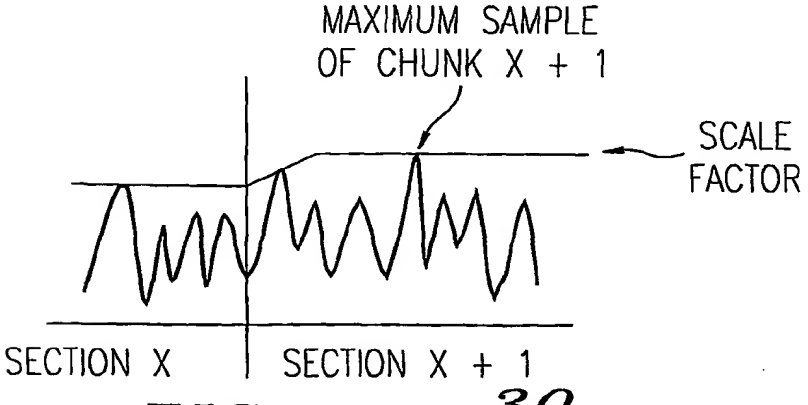


FIG. 30

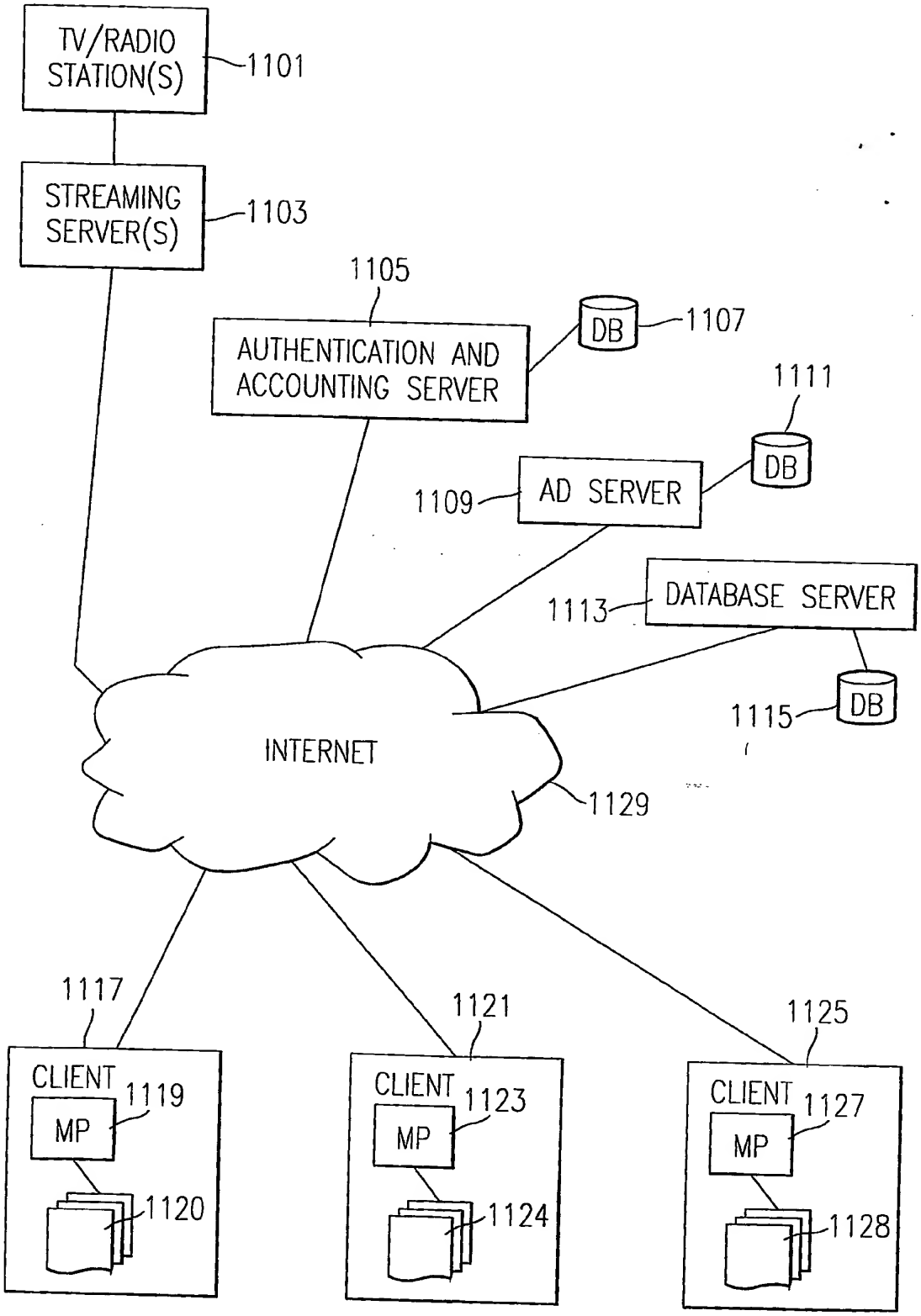

TV/RADIO STATION(S)
1101
STREAMING SERVER(S)
1103
1105
AUTHENTICATION AND ACCOUNTING SERVER
DB
1107
1111
DB
1109
AD SERVER
1113
DATABASE SERVER
DB
1115
INTERNET
1129
1117
CLIENT
MP
1119
1120
1121
CLIENT
MP
1123
1124
1125
CLIENT
MP
1127
1128

FIG. 31

1201

1203 USER NAME	1205 PASSWORD

FIG. 32

1301

1303 USER NAME	1305 GROUP ID

FIG. 33

1401

1403 USER NAME	USER PROFILE				
	1407 AGE	1409 GENDER	1411 ZIP CODE	1413 PROFESSION	1415 INCOME

FIG. 34

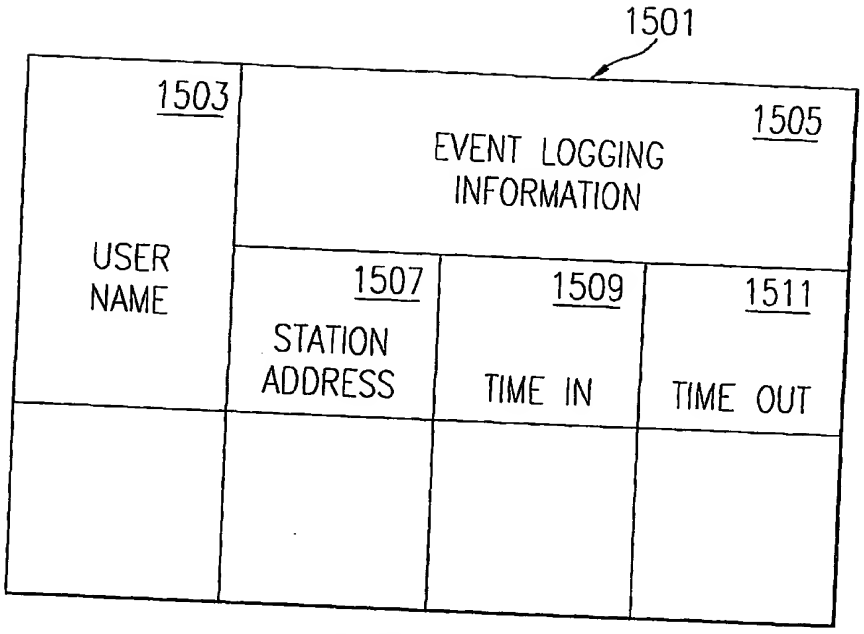


FIG. 35

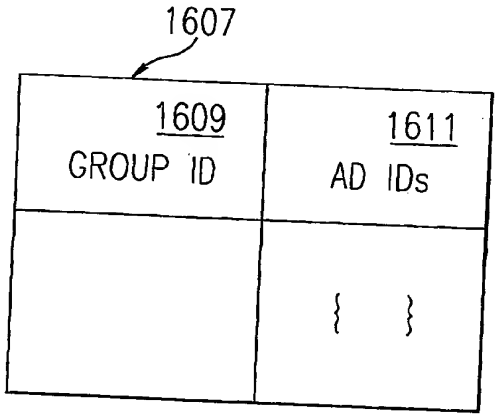


FIG. 36

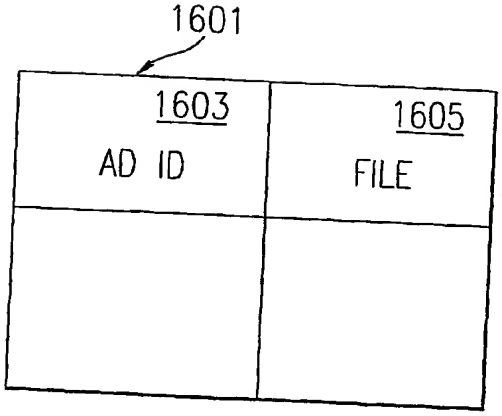


FIG. 37

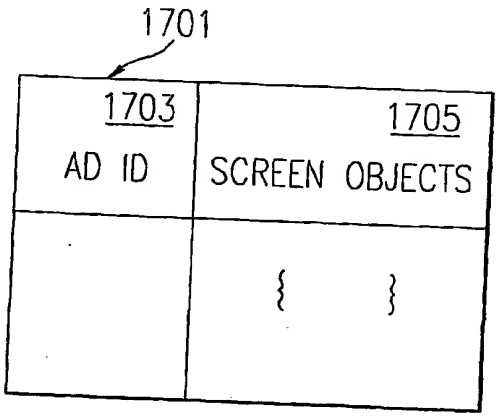
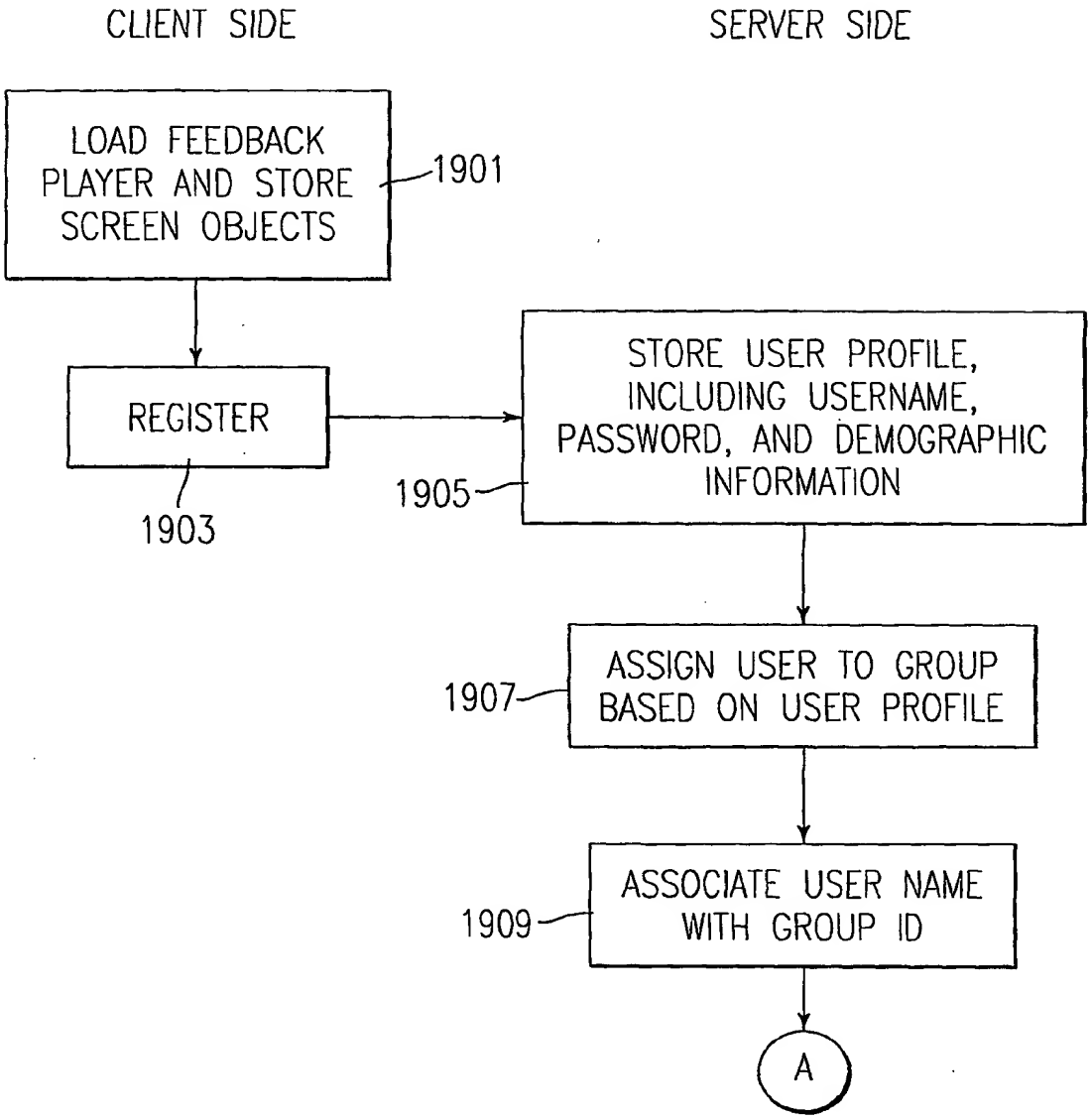


FIG. 38

FIG. 39


CLIENT SIDE
SERVER SIDE
LOAD FEEDBACK PLAYER AND STORE SCREEN OBJECTS
1901
REGISTER
1903
STORE USER PROFILE, INCLUDING USERNAME, PASSWORD, AND DEMOGRAPHIC INFORMATION
1905
ASSIGN USER TO GROUP BASED ON USER PROFILE
1907
ASSOCIATE USER NAME WITH GROUP ID
1909
A

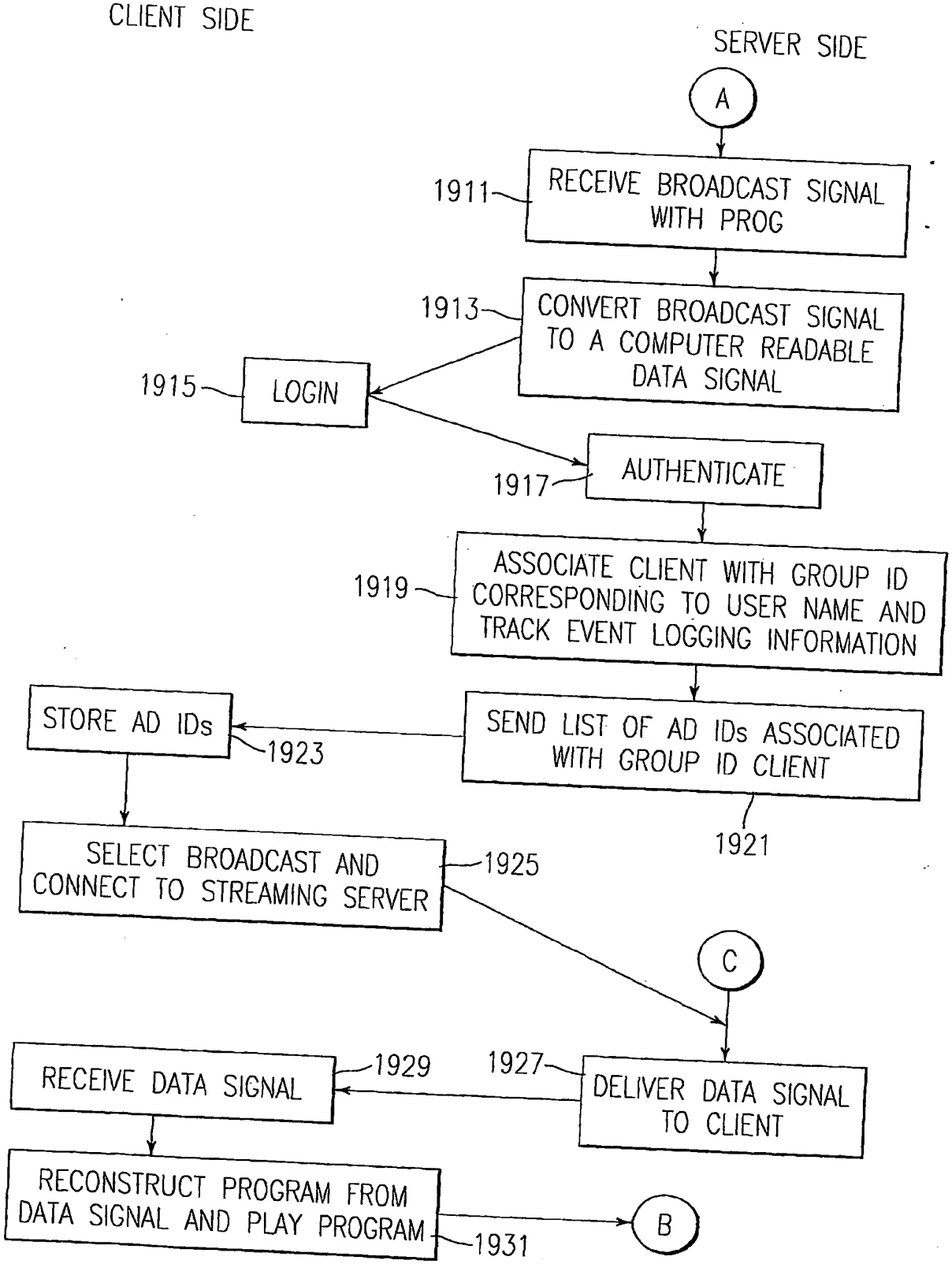

CLIENT SIDE
SERVER SIDE
A
1911
RECEIVE BROADCAST SIGNAL WITH PROG
1913
CONVERT BROADCAST SIGNAL TO A COMPUTER READABLE DATA SIGNAL
1915
LOGIN
1917
AUTHENTICATE
1919
ASSOCIATE CLIENT WITH GROUP ID CORRESPONDING TO USER NAME AND TRACK EVENT LOGGING INFORMATION
STORE AD IDs
1923
SEND LIST OF AD IDs ASSOCIATED WITH GROUP ID CLIENT
1921
SELECT BROADCAST AND CONNECT TO STREAMING SERVER
1925
C
RECEIVE DATA SIGNAL
1929
1927
DELIVER DATA SIGNAL TO CLIENT
RECONSTRUCT PROGRAM FROM DATA SIGNAL AND PLAY PROGRAM
1931
B

FIG. 40

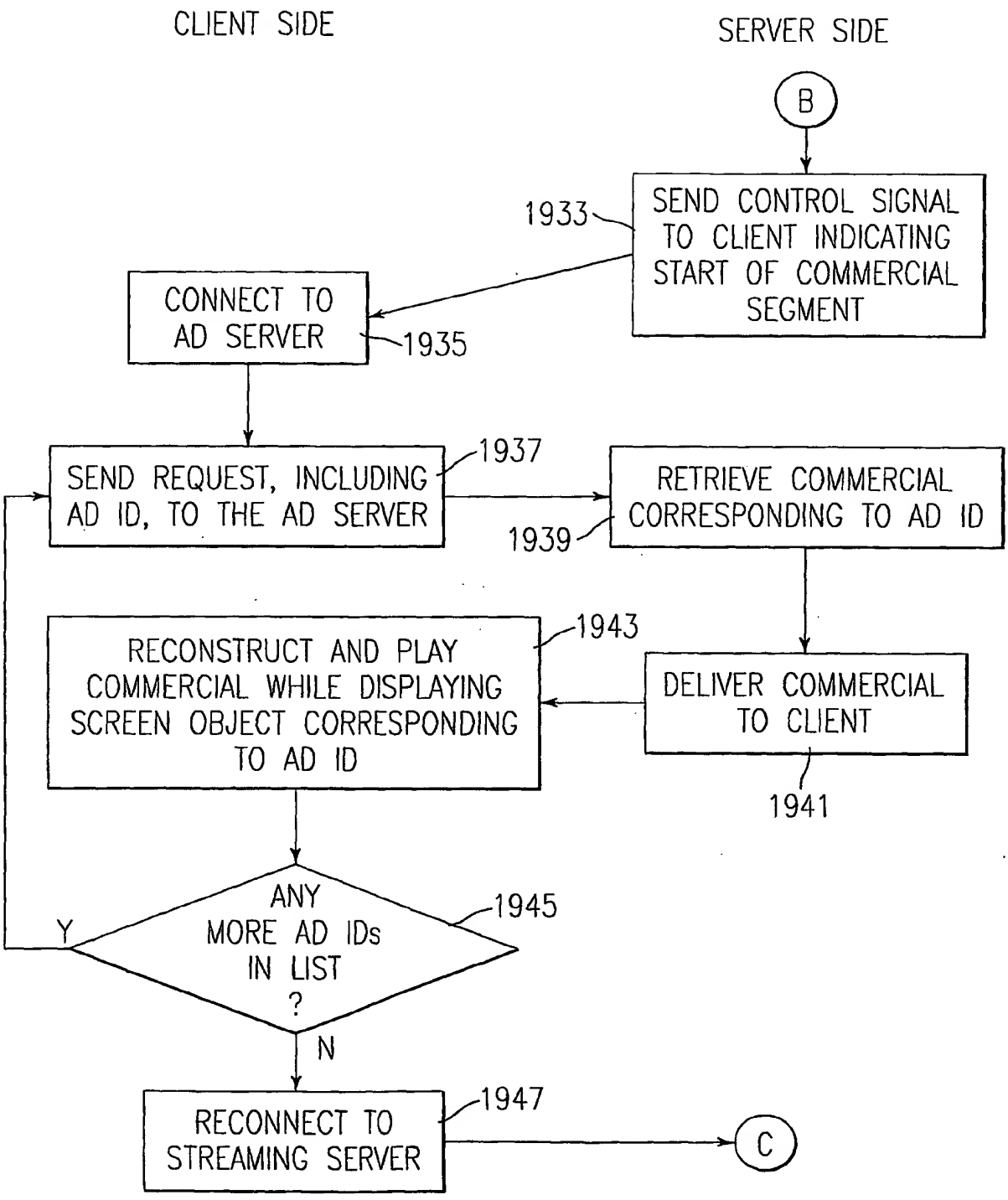

CLIENT SIDE
SERVER SIDE
B
SEND CONTROL SIGNAL TO CLIENT INDICATING START OF COMMERCIAL SEGMENT
1933
CONNECT TO AD SERVER
1935
SEND REQUEST, INCLUDING AD ID, TO THE AD SERVER
1937
RETRIEVE COMMERCIAL CORRESPONDING TO AD ID
1939
RECONSTRUCT AND PLAY COMMERCIAL WHILE DISPLAYING SCREEN OBJECT CORRESPONDING TO AD ID
1943
DELIVER COMMERCIAL TO CLIENT
1941
ANY MORE AD IDs IN LIST ?
1945
Y
N
RECONNECT TO STREAMING SERVER
1947
C

FIG. 41

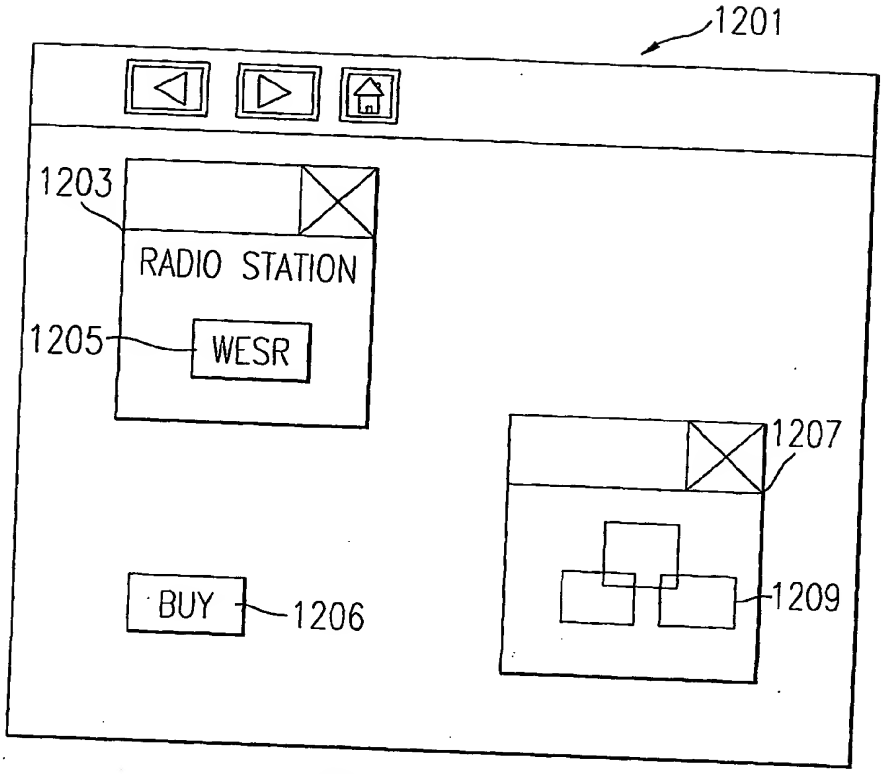


FIG. 42

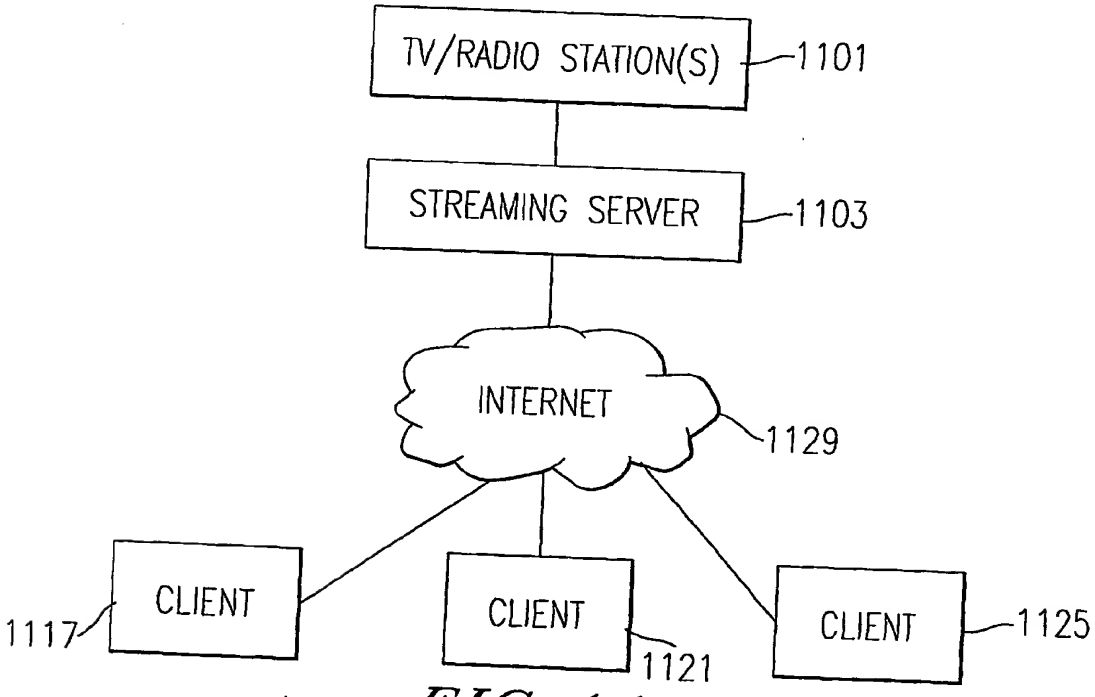


FIG. 44
PRIOR ART

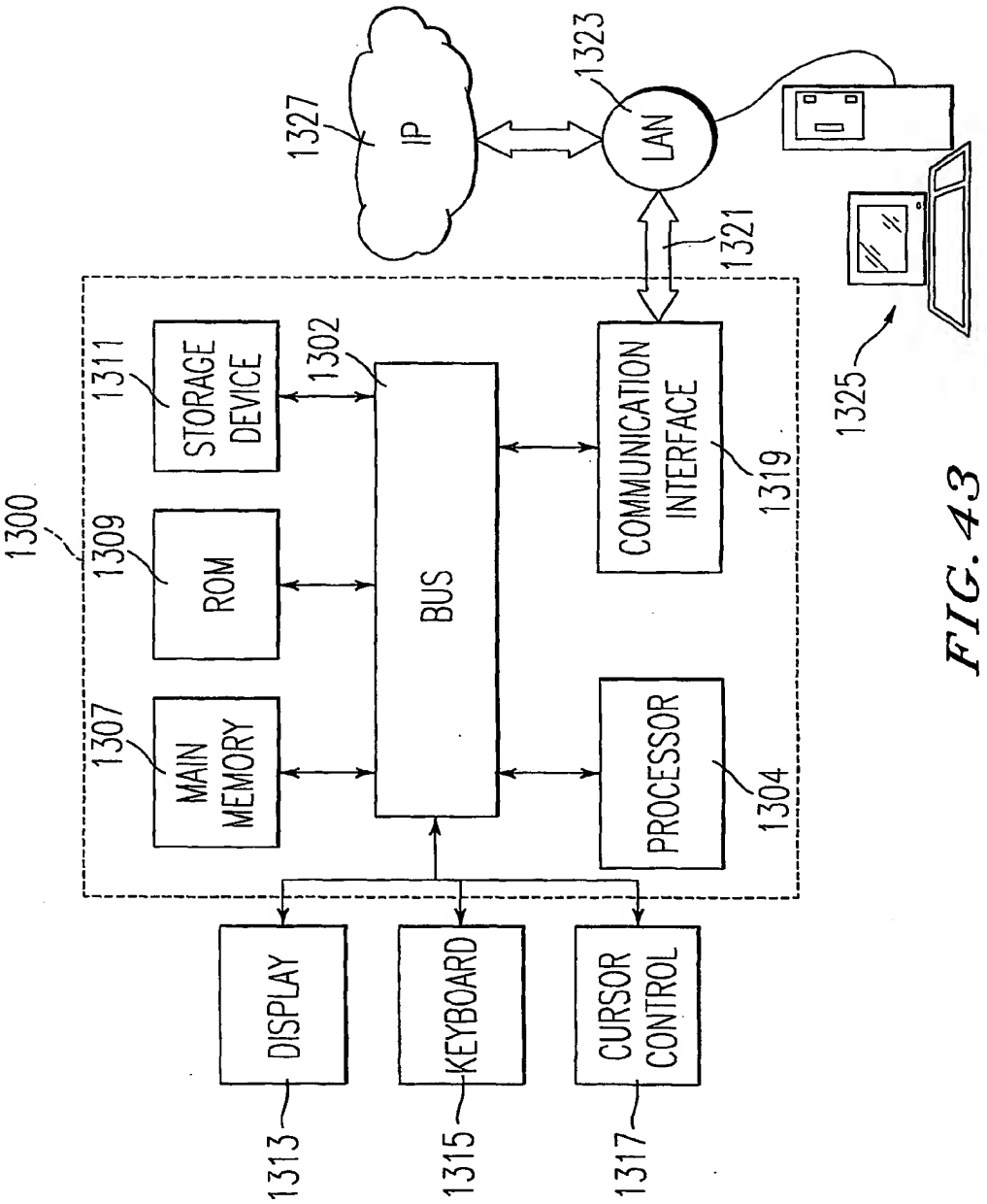
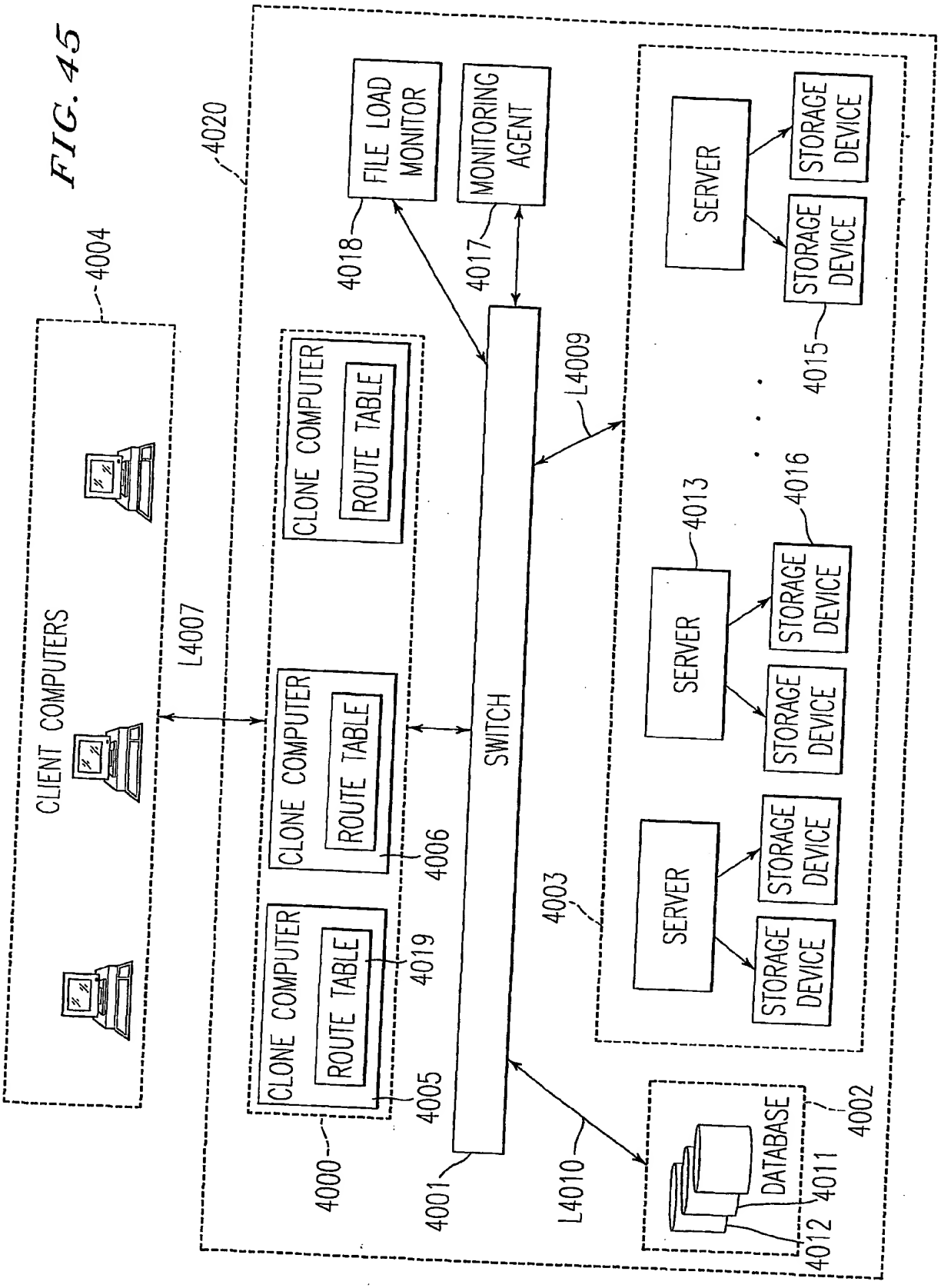


FIG. 43

FIG. 45


4004
CLIENT COMPUTERS
L4007
4020
FILE LOAD MONITOR
4018
MONITORING AGENT
4017
CLONE COMPUTER
ROUTE TABLE
4000
CLONE COMPUTER
ROUTE TABLE
4005
CLONE COMPUTER
ROUTE TABLE
4019
4006
SWITCH
4001
L4009
L4010
4003
SERVER
STORAGE DEVICE
STORAGE DEVICE
4015
SERVER
4013
STORAGE DEVICE
4016
STORAGE DEVICE
SERVER
STORAGE DEVICE
STORAGE DEVICE
DATABASE
4012
4011
4002

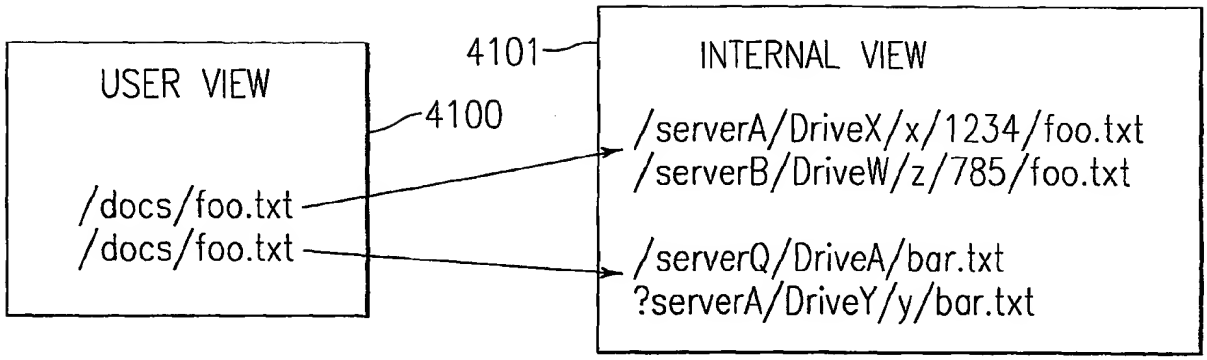

USER VIEW
4100
/docs/foo.txt
/docs/foo.txt
4101
INTERNAL VIEW
/serverA/DriveX/x/1234/foo.txt
/serverB/DriveW/z/785/foo.txt
/serverQ/DriveA/bar.txt
?serverA/DriveY/y/bar.txt

FIG. 46

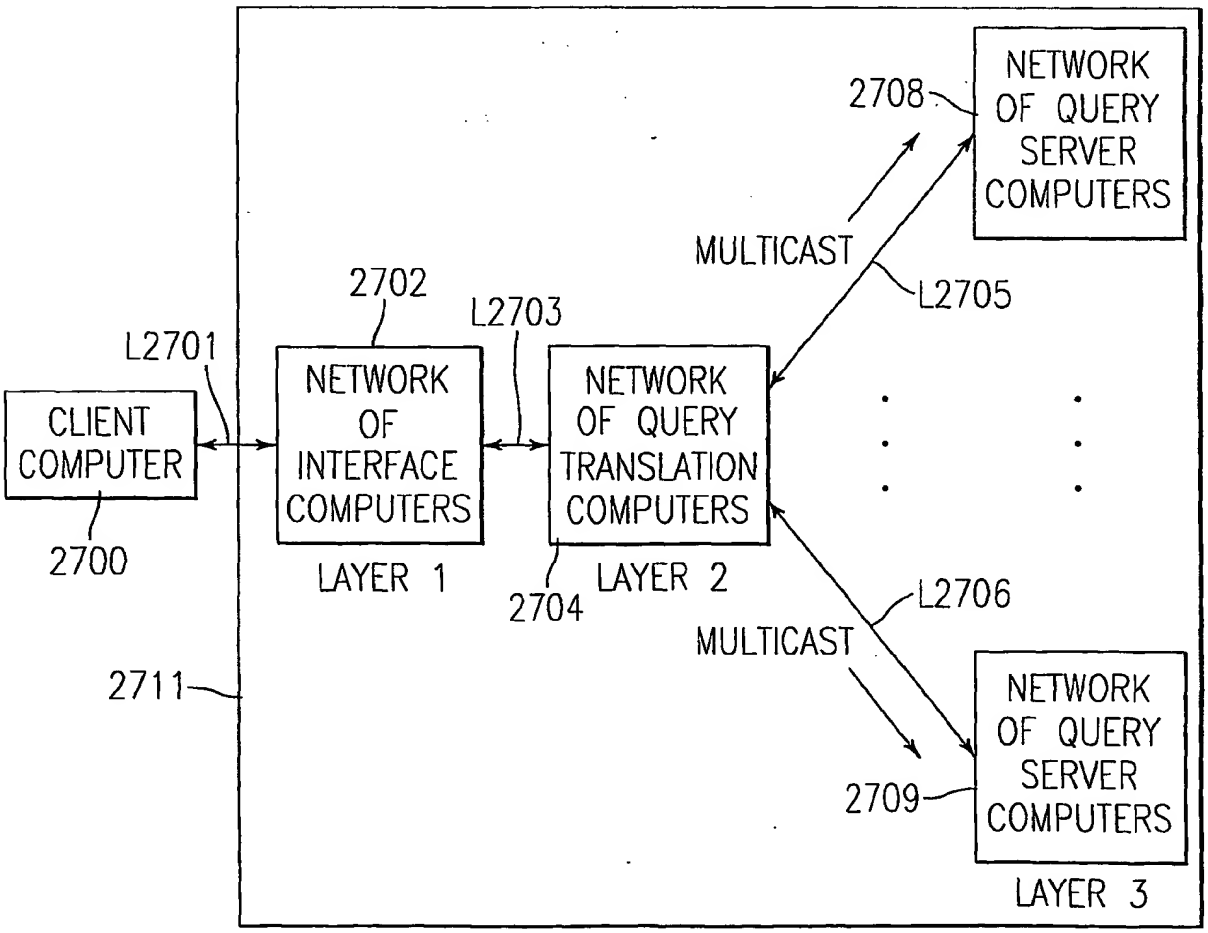

2708
NETWORK OF QUERY SERVER COMPUTERS
MULTICAST
L2705
2702
L2703
L2701
NETWORK OF INTERFACE COMPUTERS
NETWORK OF QUERY TRANSLATION COMPUTERS
CLIENT COMPUTER
2700
LAYER 1
2704
LAYER 2
L2706
MULTICAST
2711
NETWORK OF QUERY SERVER COMPUTERS
2709
LAYER 3

FIG. 47

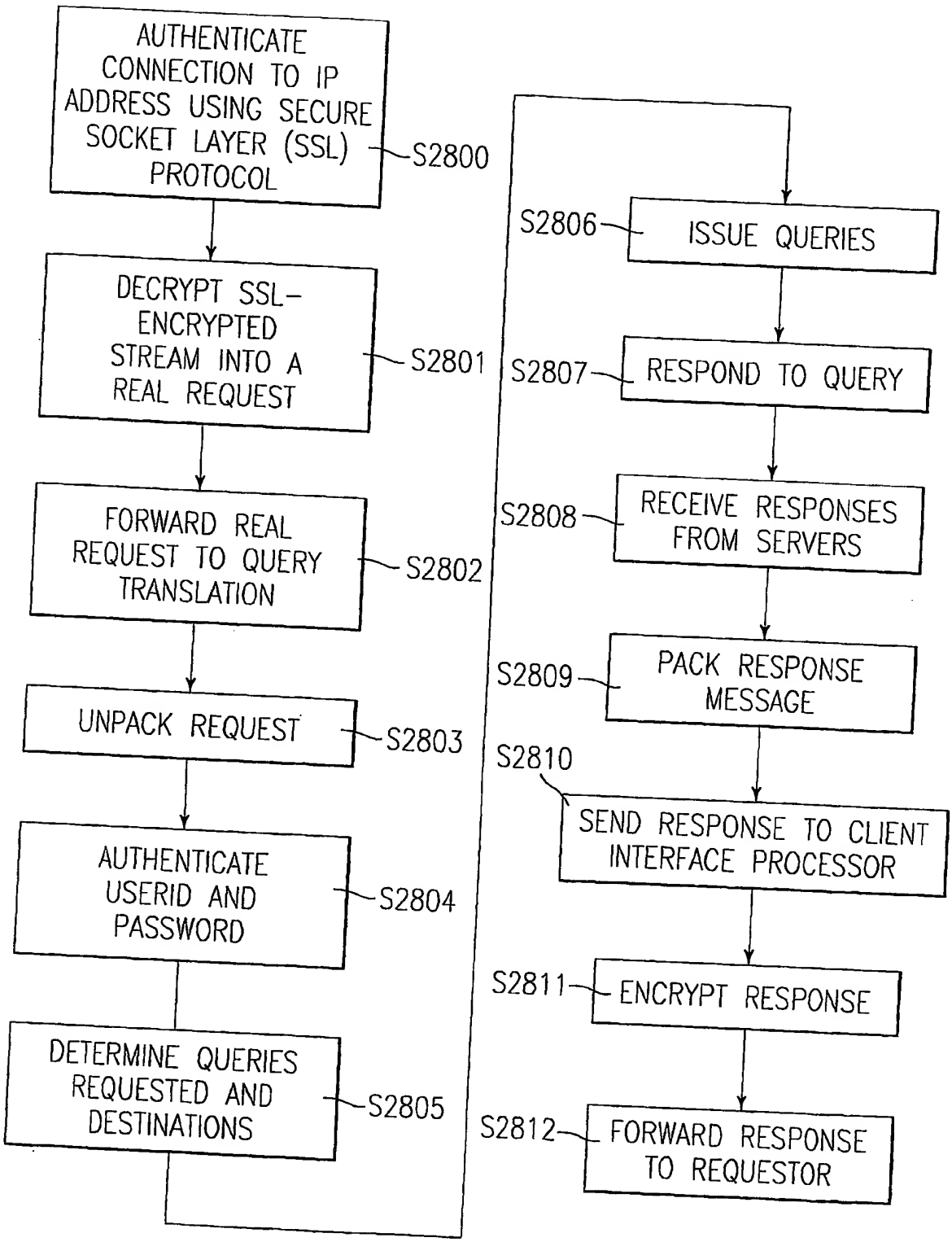


FIG. 48

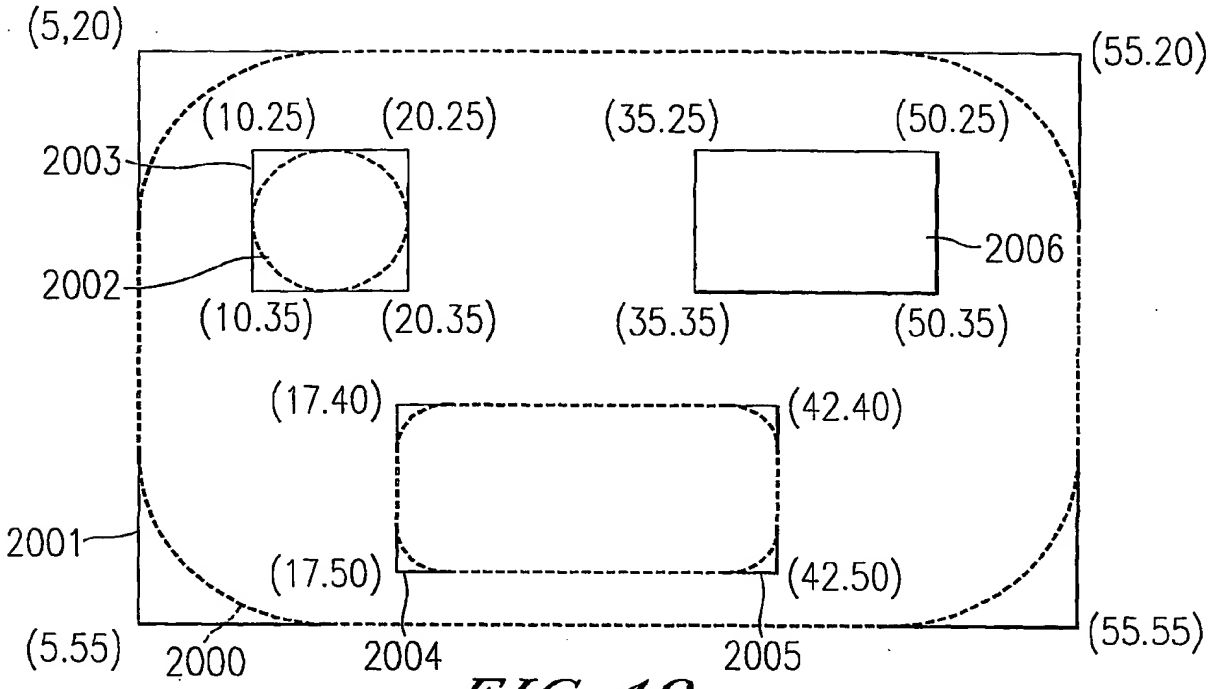


FIG. 49

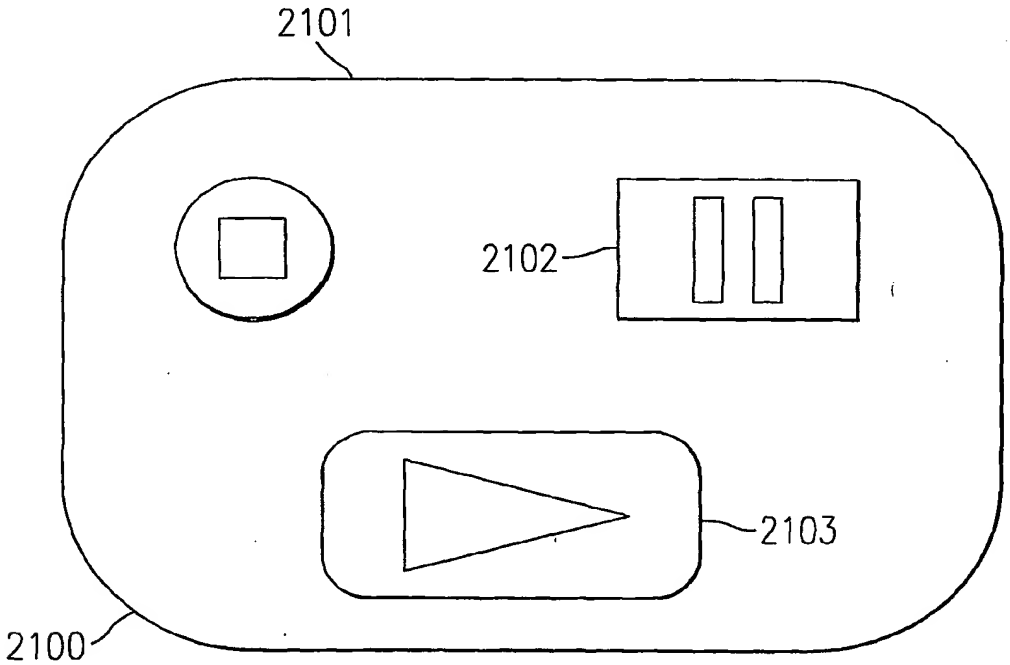


FIG. 50

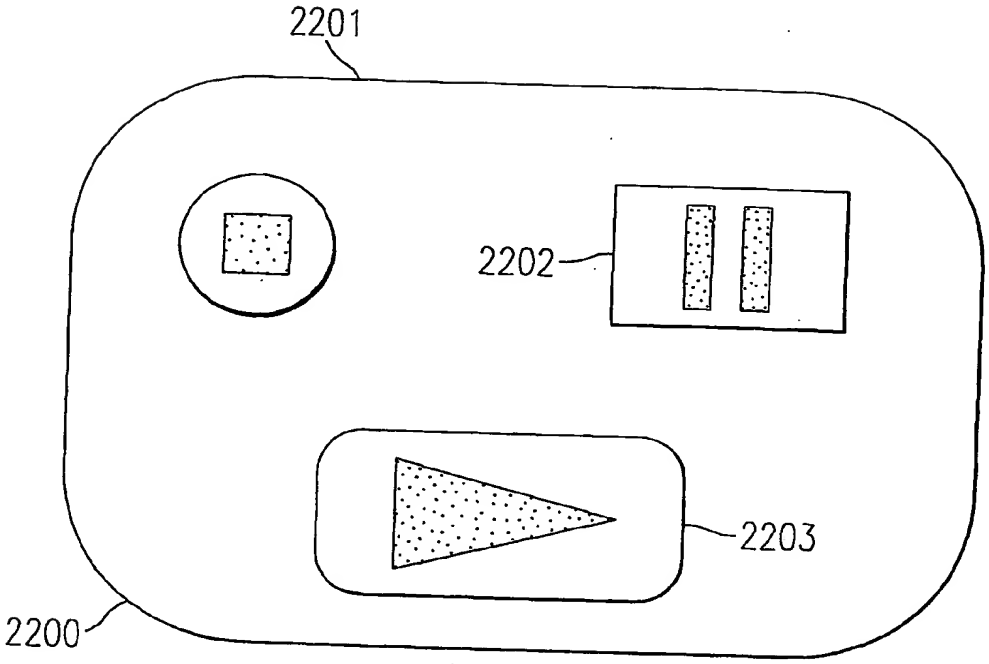


FIG. 51

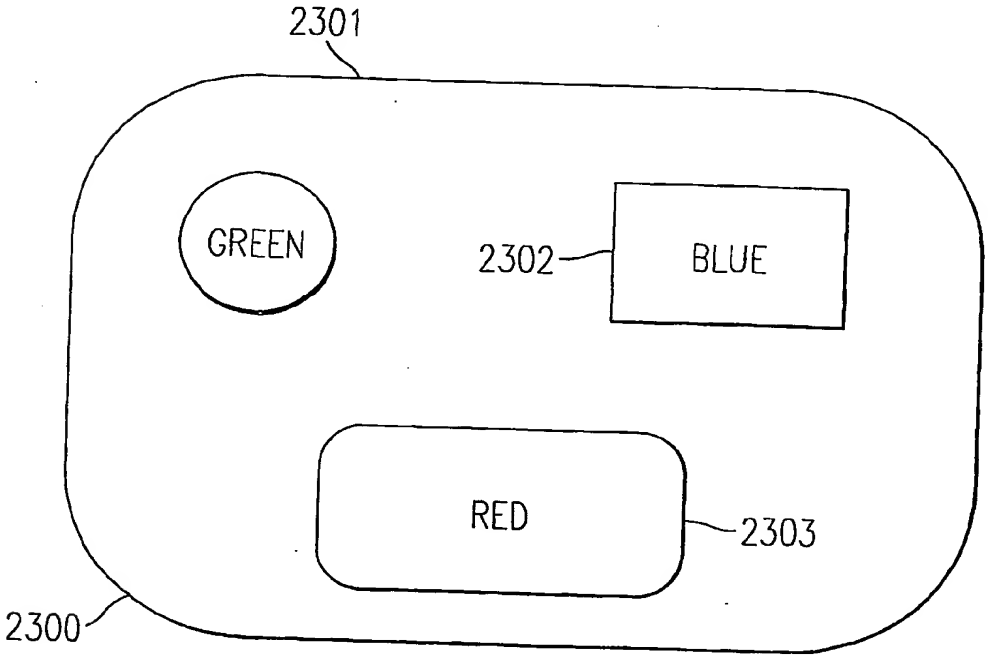


FIG. 52

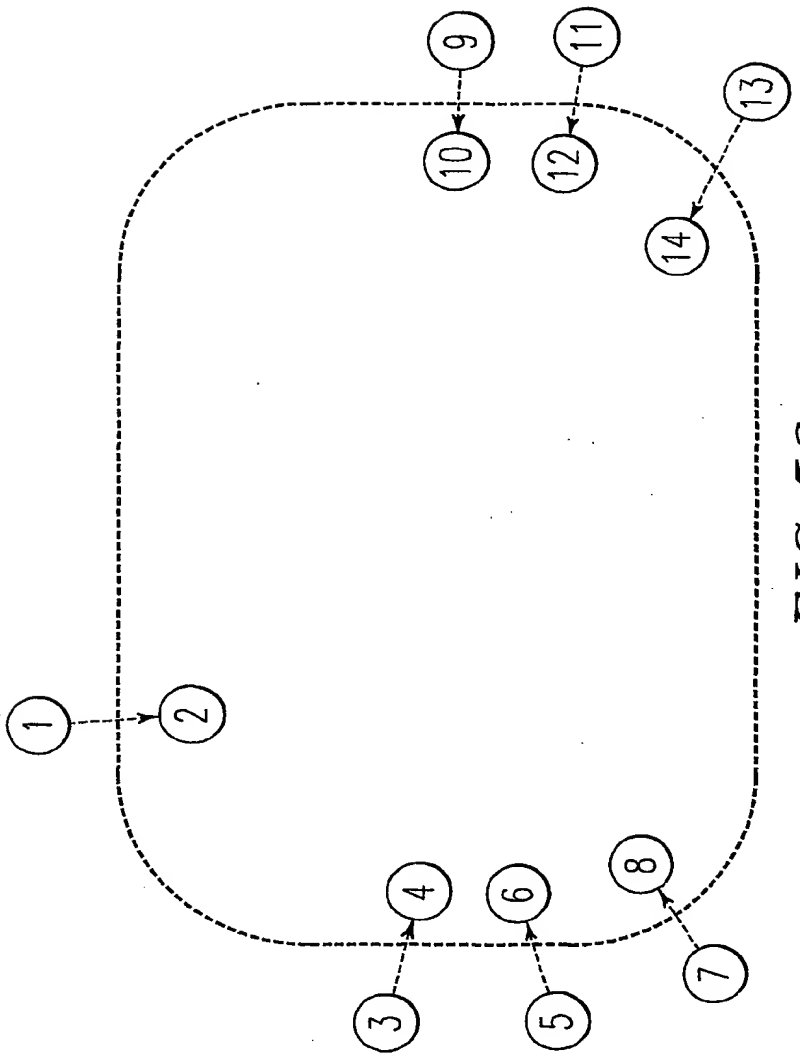


FIG. 53

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9 ! @ # $ % ^ & * ( ) _ + - = , . < > / \ ; : " ' [ ]

FIG. 54

FIG. 55

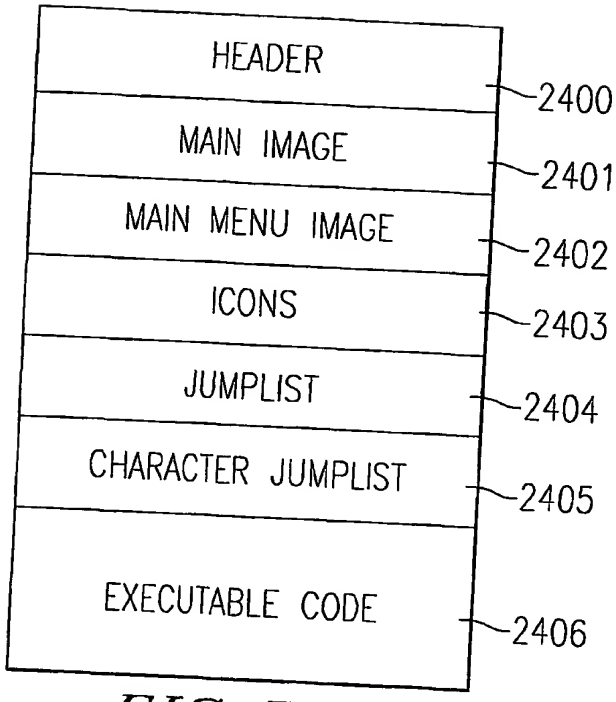


FIG. 56

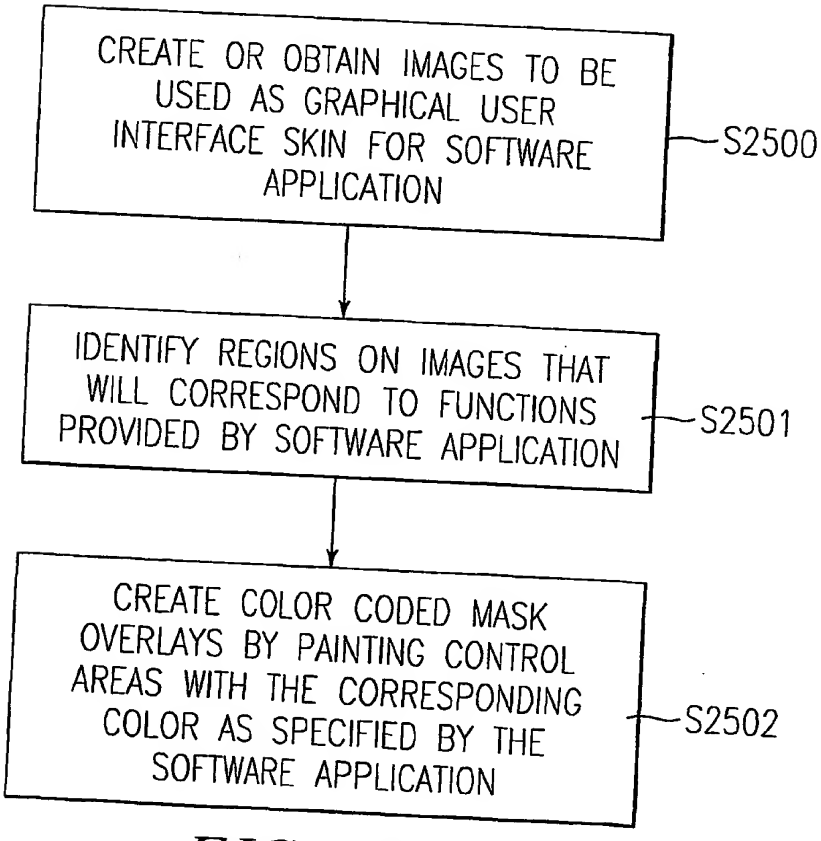


FIG. 57

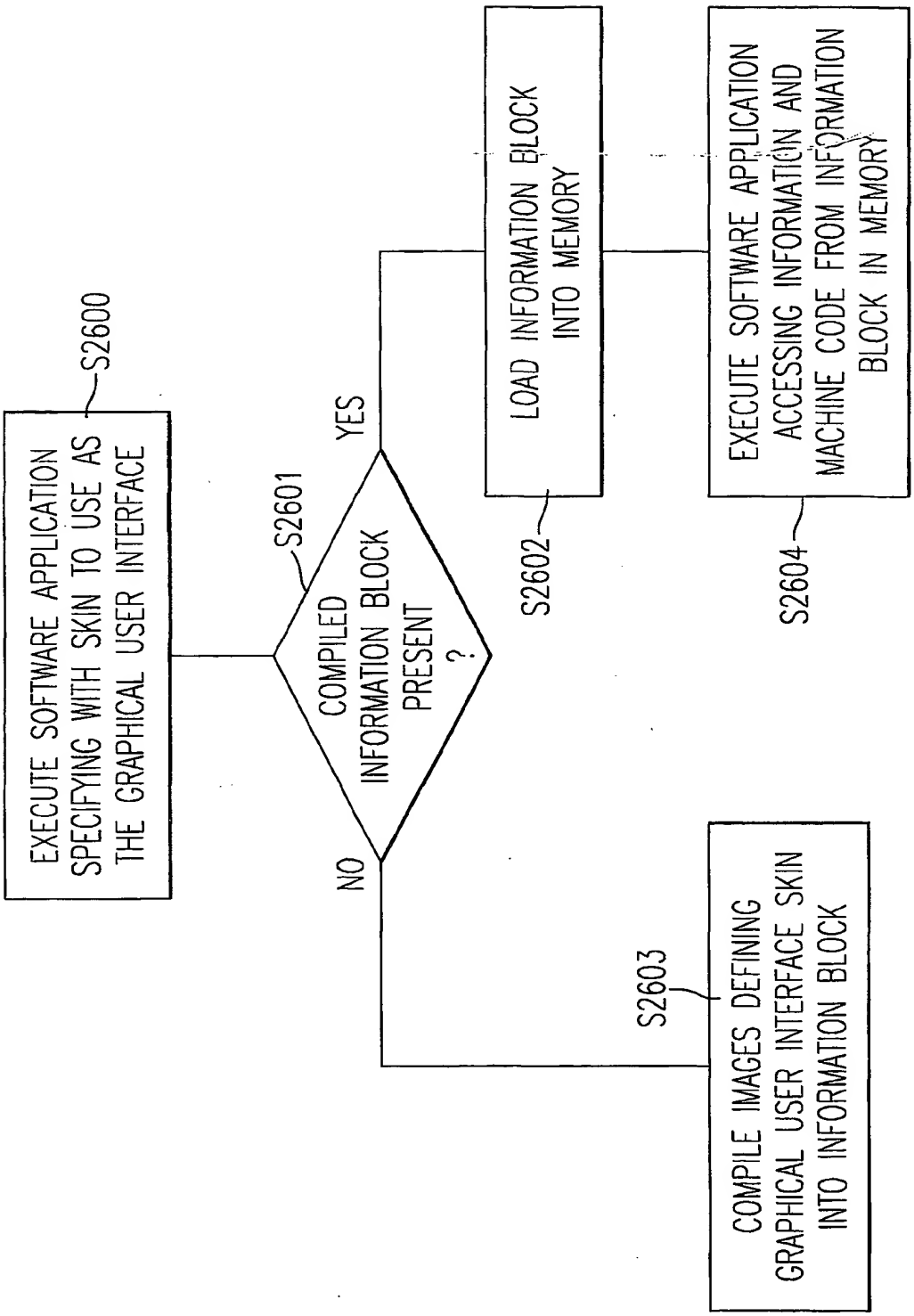

EXECUTE SOFTWARE APPLICATION SPECIFYING WITH SKIN TO USE AS THE GRAPHICAL USER INTERFACE
S2600
COMPILED INFORMATION BLOCK PRESENT ?
S2601
YES
NO
LOAD INFORMATION BLOCK INTO MEMORY
S2602
EXECUTE SOFTWARE APPLICATION ACCESSING INFORMATION AND MACHINE CODE FROM INFORMATION BLOCK IN MEMORY
S2604
COMPILE IMAGES DEFINING GRAPHICAL USER INTERFACE SKIN INTO INFORMATION BLOCK
S2603

FIG. 58

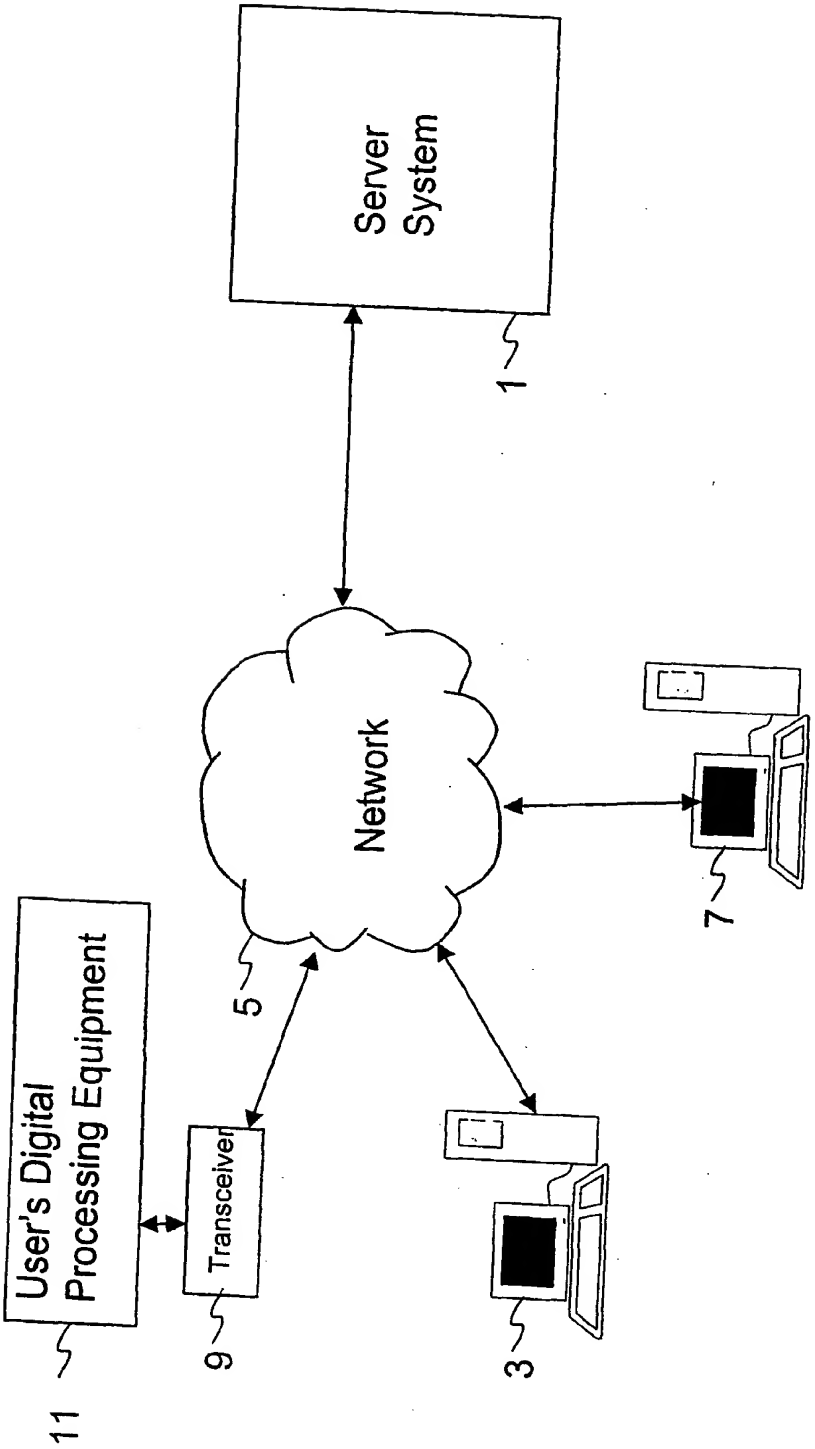

Server System
1
Network
5
7
User's Digital Processing Equipment
11
Transceiver
9
3

FIG. 1

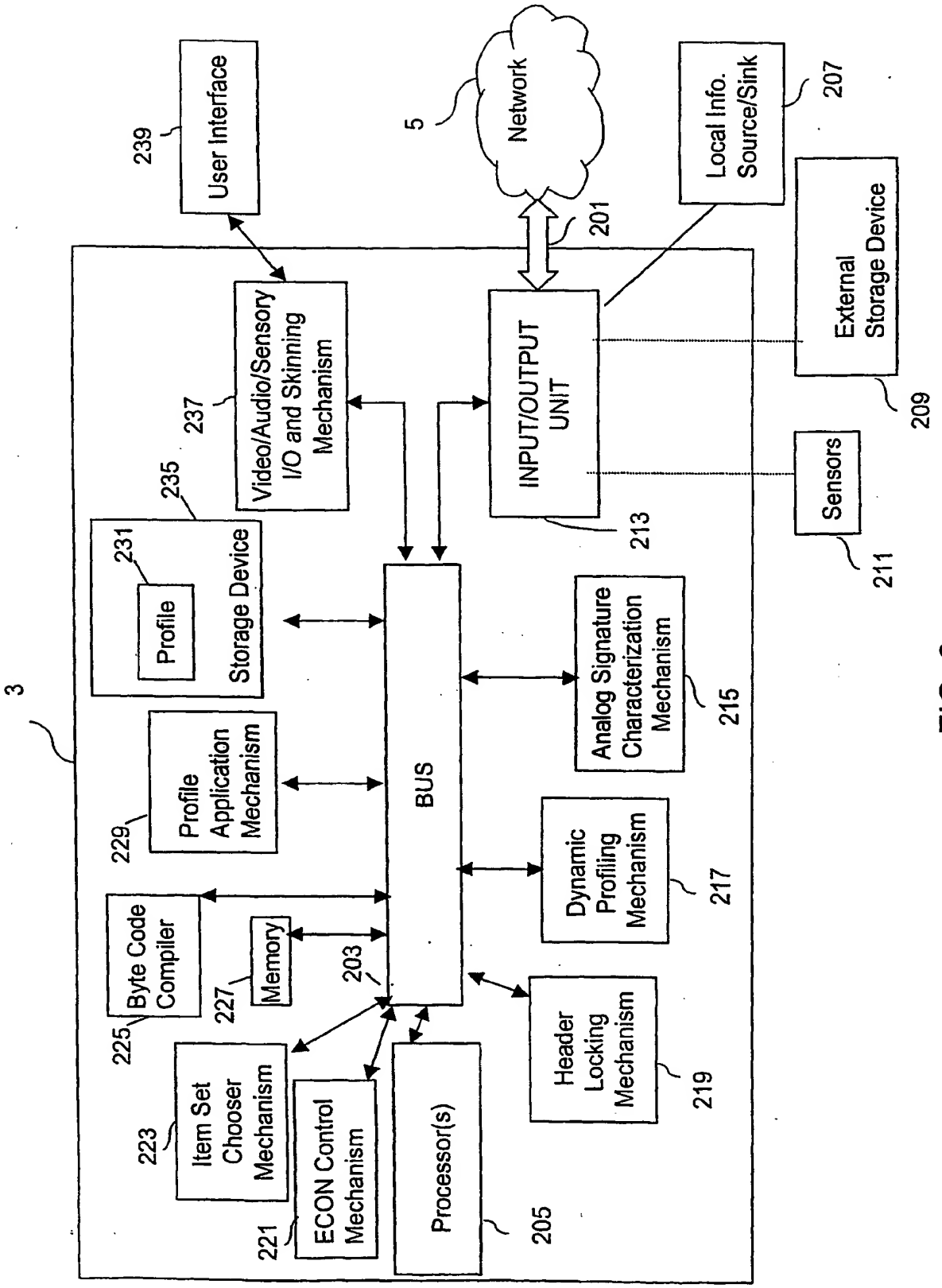

User Interface
239
Network
5
Local Info. Source/Sink
207
External Storage Device
209
201
Video/Audio/Sensory I/O and Skinning Mechanism
237
INPUT/OUTPUT UNIT
213
Sensors
211
235
Storage Device
231
Profile
3
BUS
Analog Signature Characterization Mechanism
215
Profile Application Mechanism
229
Dynamic Profiling Mechanism
217
Byte Code Compiler
225
Memory
227
203
Header Locking Mechanism
219
Item Set Chooser Mechanism
223
ECON Control Mechanism
221
Processor(s)
205

FIG. 2

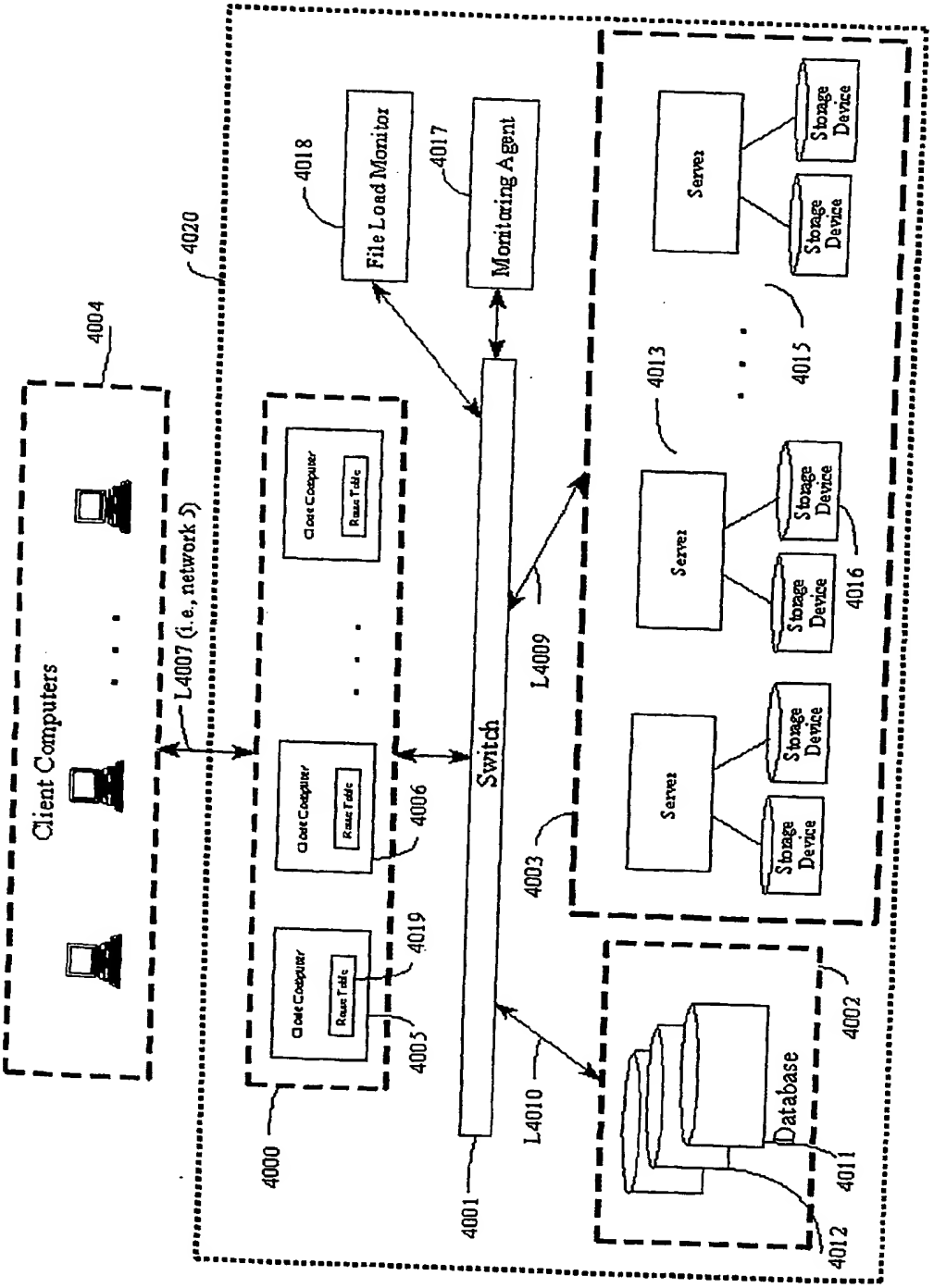

Client Computers
4004
L4007 (i.e., network 5)
4020
4018
File Load Monitor
4017
Monitoring Agent
Client Computer
Route Table
4006
4019
4005
4000
Switch
L4009
L4010
4001
4013
4015
4016
4003
Server
Storage Device
4002
Database
4011
4012

Figure 3

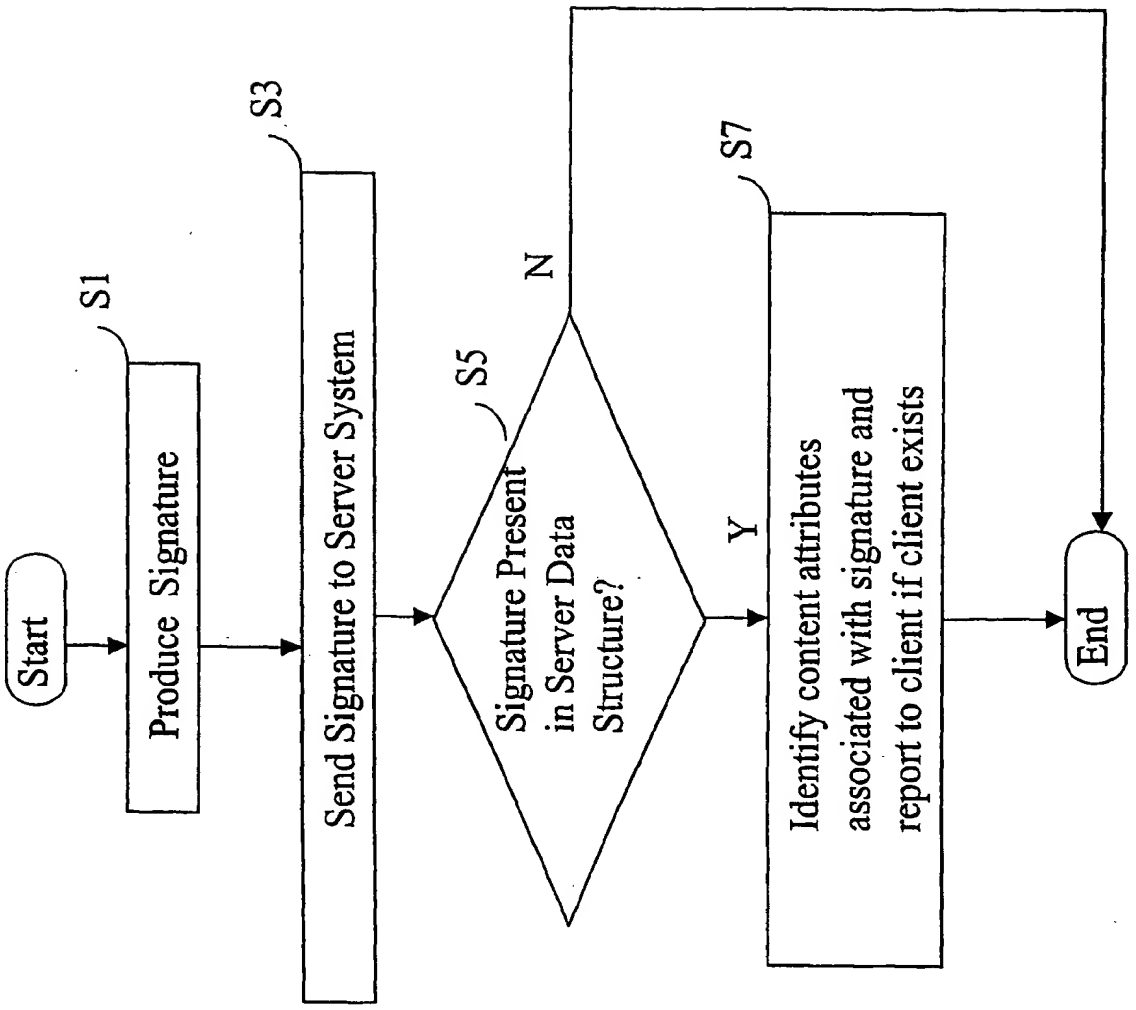

Start
S1
Produce Signature
S3
Send Signature to Server System
S5
Signature Present in Server Data Structure?
N
Y
S7
Identify content attributes associated with signature and report to client if client exists
End

FIG. 4

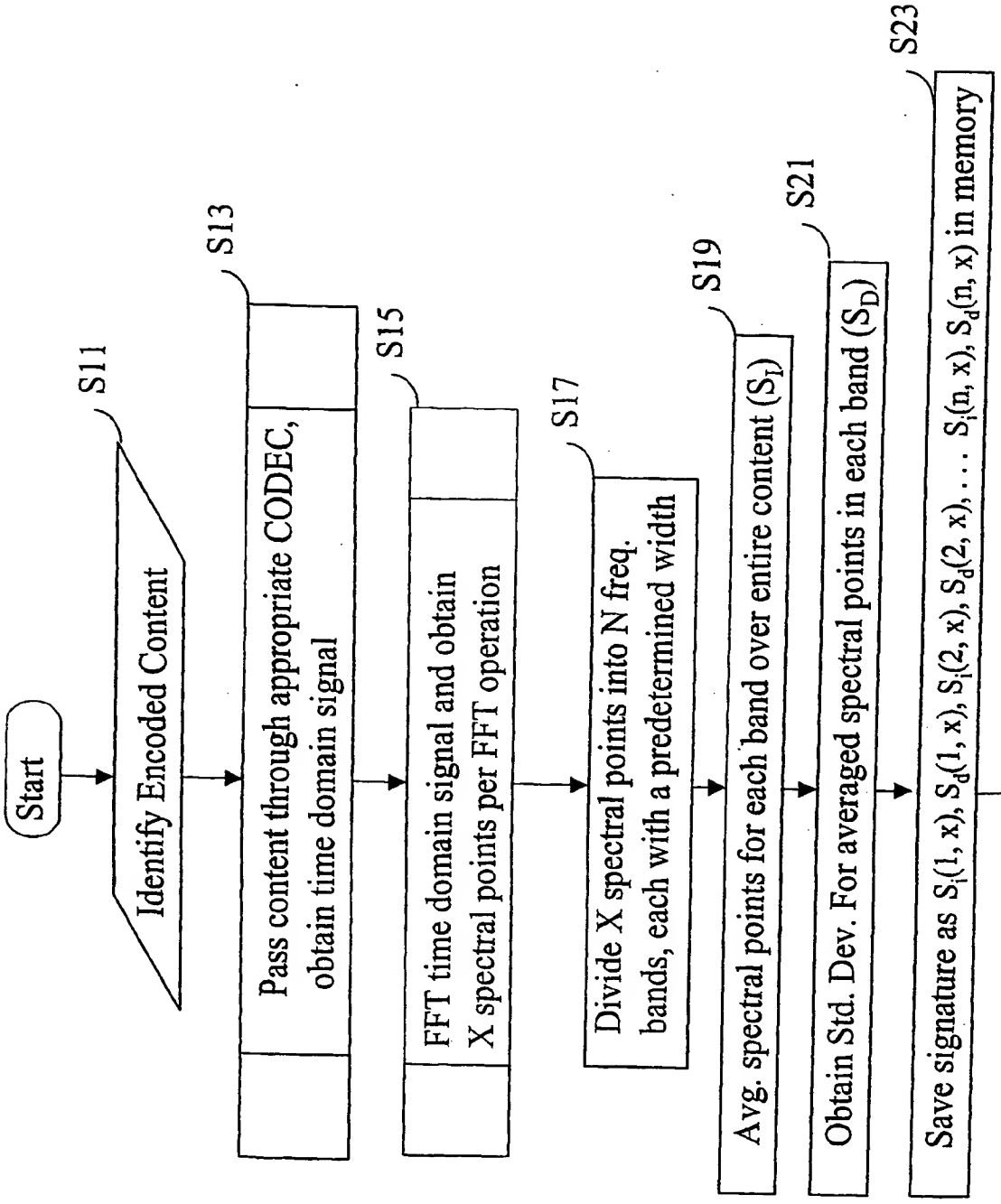


FIG. 5

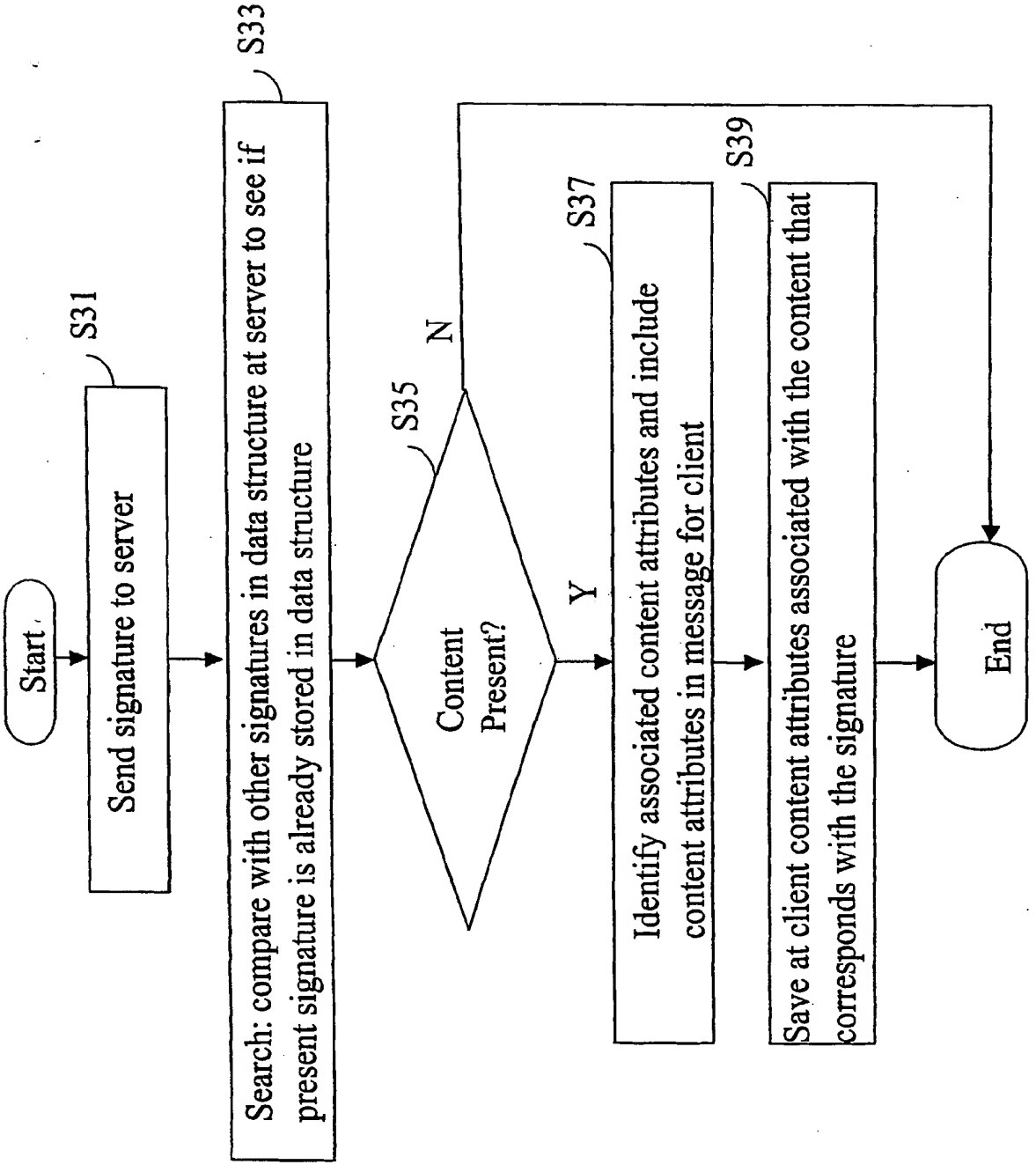

Start
S31
Send signature to server
S33
Search: compare with other signatures in data structure at server to see if present signature is already stored in data structure
S35
Content Present?
N
Y
S37
Identify associated content attributes and include content attributes in message for client
S39
Save at client content attributes associated with the content that corresponds with the signature
End

FIG. 6

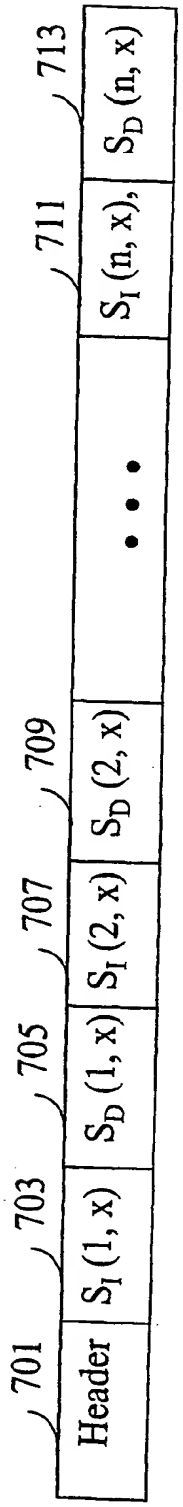

701
Header
703
S_I (1, x)
705
S_D (1, x)
707
S_I (2, x)
709
S_D (2, x)
711
S_I (n, x),
713
S_D (n, x)

FIG. 7

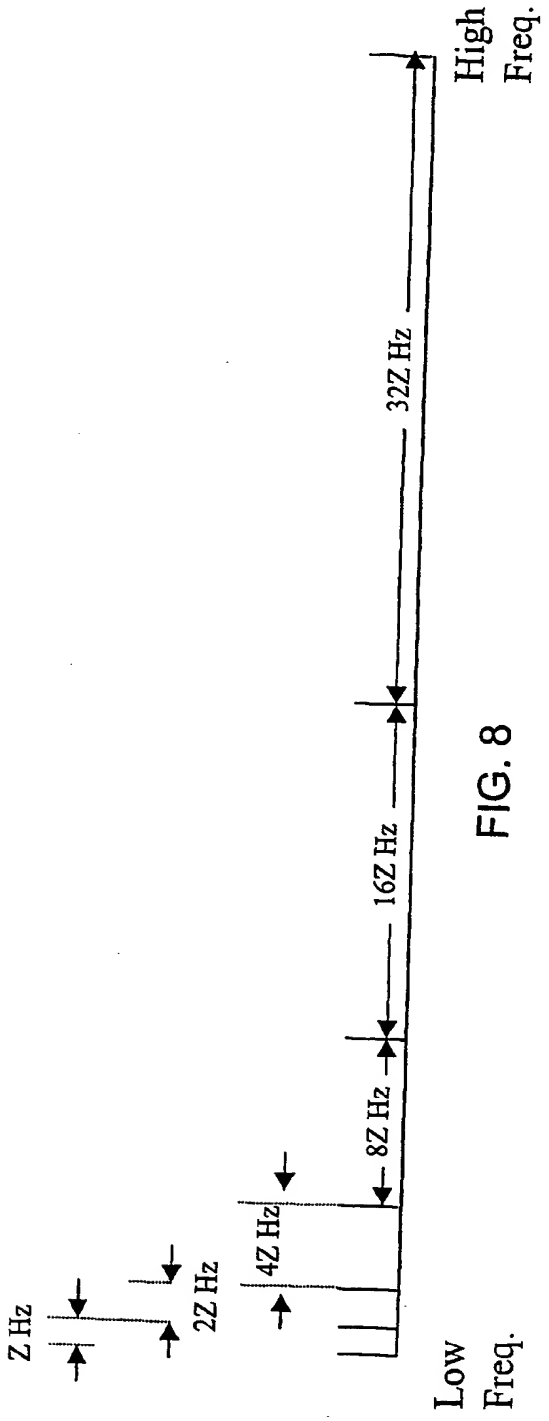
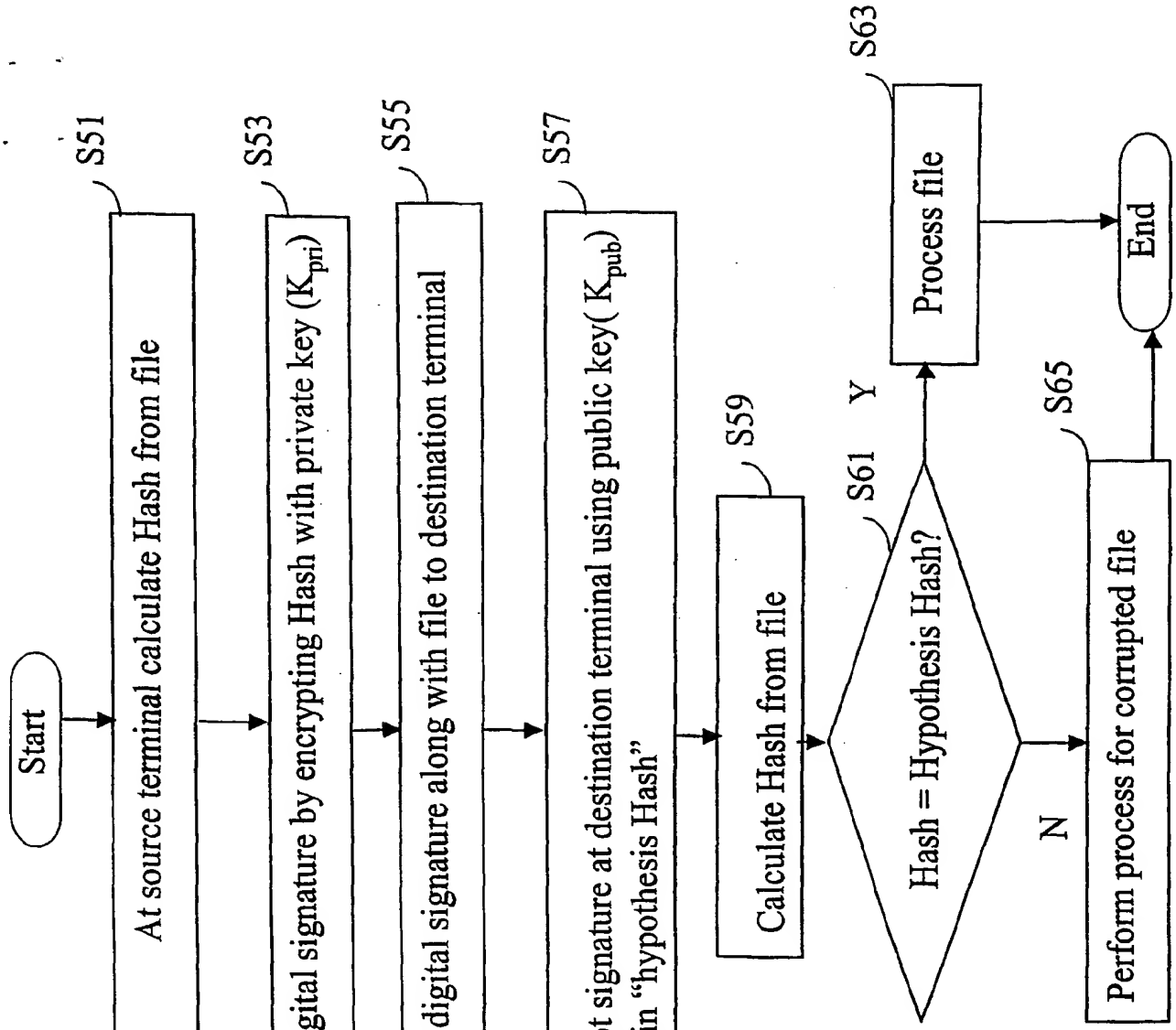

Z Hz
2Z Hz
4Z Hz
8Z Hz
16Z Hz
32Z Hz
Low Freq.
High Freq.

FIG. 8



Background Art
FIG. 9

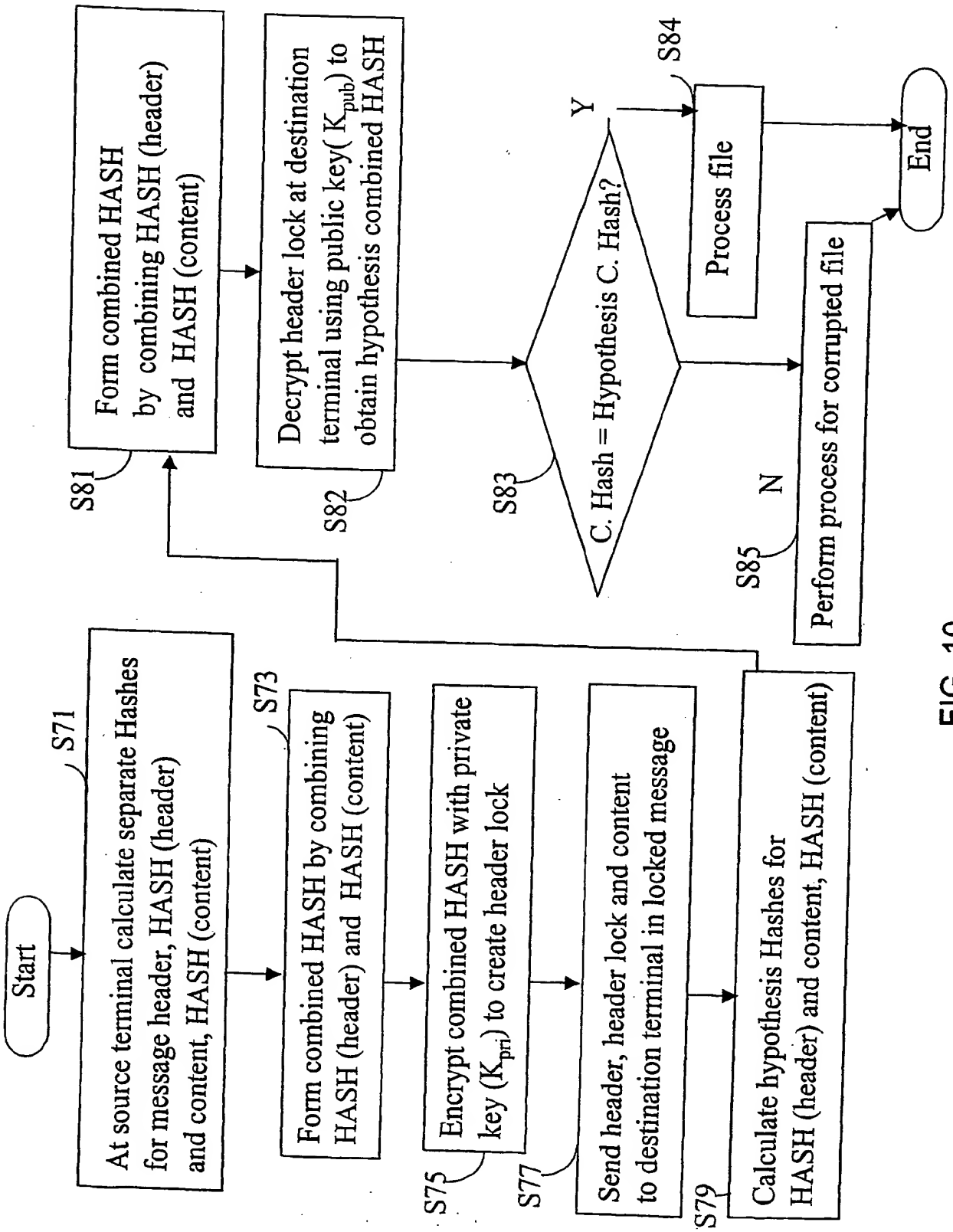

Start
S71
At source terminal calculate separate Hashes for message header, HASH (header) and content, HASH (content)
S73
Form combined HASH by combining HASH (header) and HASH (content)
S75
Encrypt combined HASH with private key (K_{pri}) to create header lock
S77
Send header, header lock and content to destination terminal in locked message
S79
Calculate hypothesis Hashes for HASH (header) and content, HASH (content)
S81
Form combined HASH by combining HASH (header) and HASH (content)
S82
Decrypt header lock at destination terminal using public key(K_{pub}) to obtain hypothesis combined HASH
S83
C. Hash = Hypothesis C. Hash?
Y
S84
Process file
End
N
S85
Perform process for corrupted file

FIG. 10

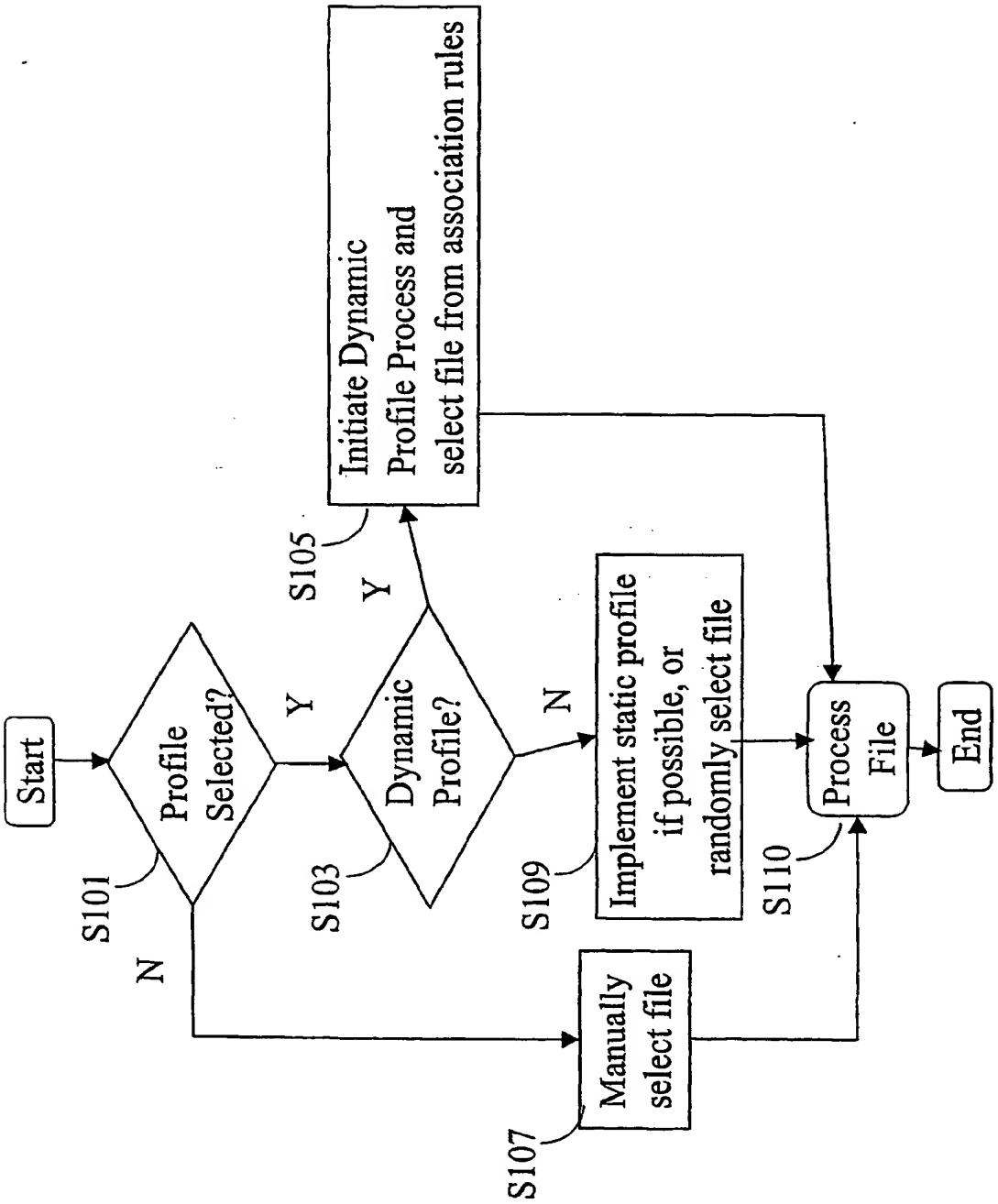
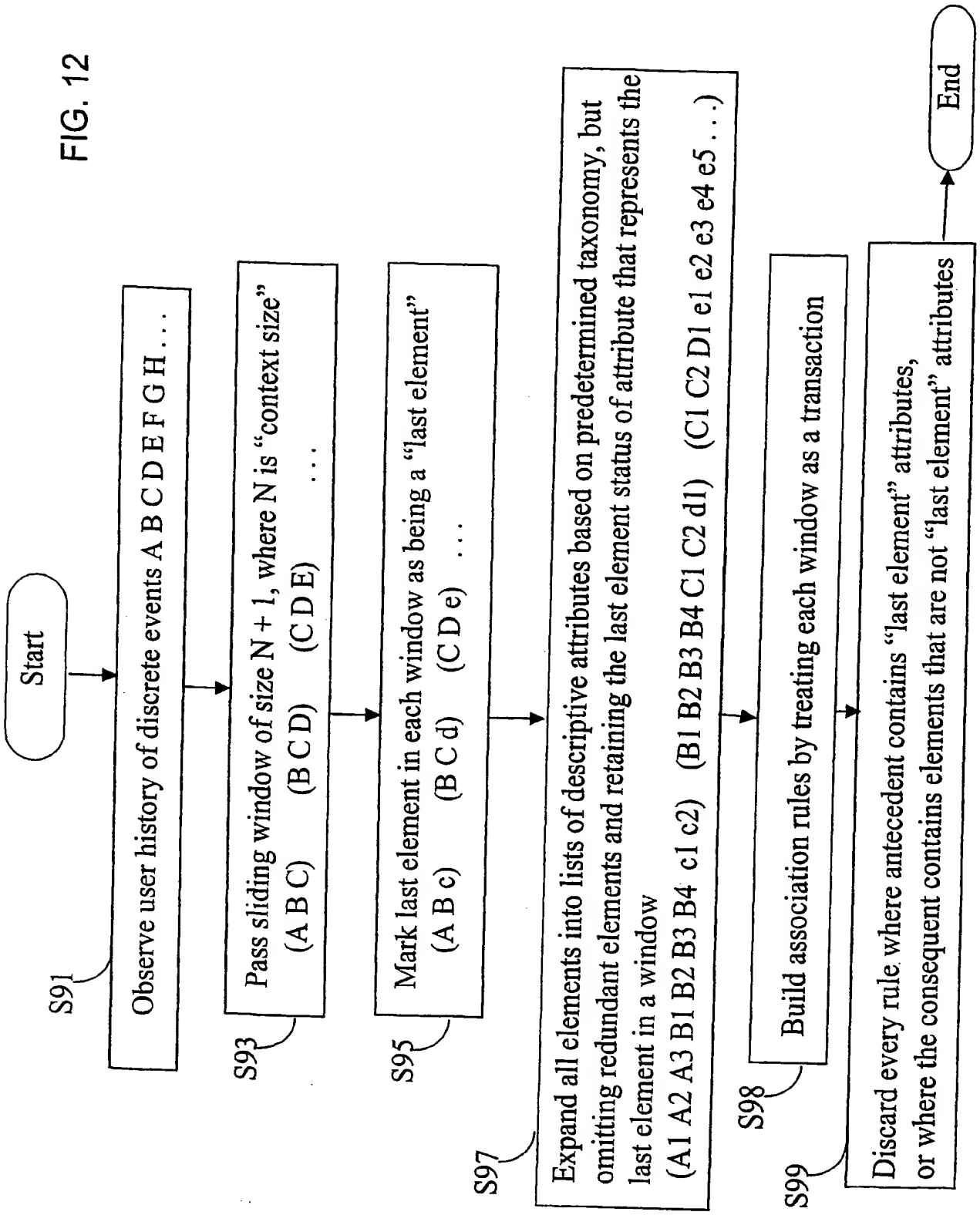

Start
Profile Selected?
S101
Y
N
Dynamic Profile?
S103
Y
N
S105
Initiate Dynamic Profile Process and select file from association rules
S109
Implement static profile if possible, or randomly select file
S107
Manually select file
S110
Process File
End

FIG. 11

FIG. 12


Start
S91
Observe user history of discrete events A B C D E F G H . . .
S93
Pass sliding window of size N + 1, where N is "context size"
(A B C) (B C D) (C D E) . . .
S95
Mark last element in each window as being a "last element"
(A B c) (B C d) (C D e) . . .
S97
Expand all elements into lists of descriptive attributes based on predetermined taxonomy, but omitting redundant elements and retaining the last element status of attribute that represents the last element in a window
(A1 A2 A3 B1 B2 B3 B4 c1 c2) (B1 B2 B3 B4 C1 C2 d1) (C1 C2 D1 e1 e2 e3 e4 e5 . . .)
S98
Build association rules by treating each window as a transaction
S99
Discard every rule where antecedent contains "last element" attributes, or where the consequent contains elements that are not "last element" attributes
End

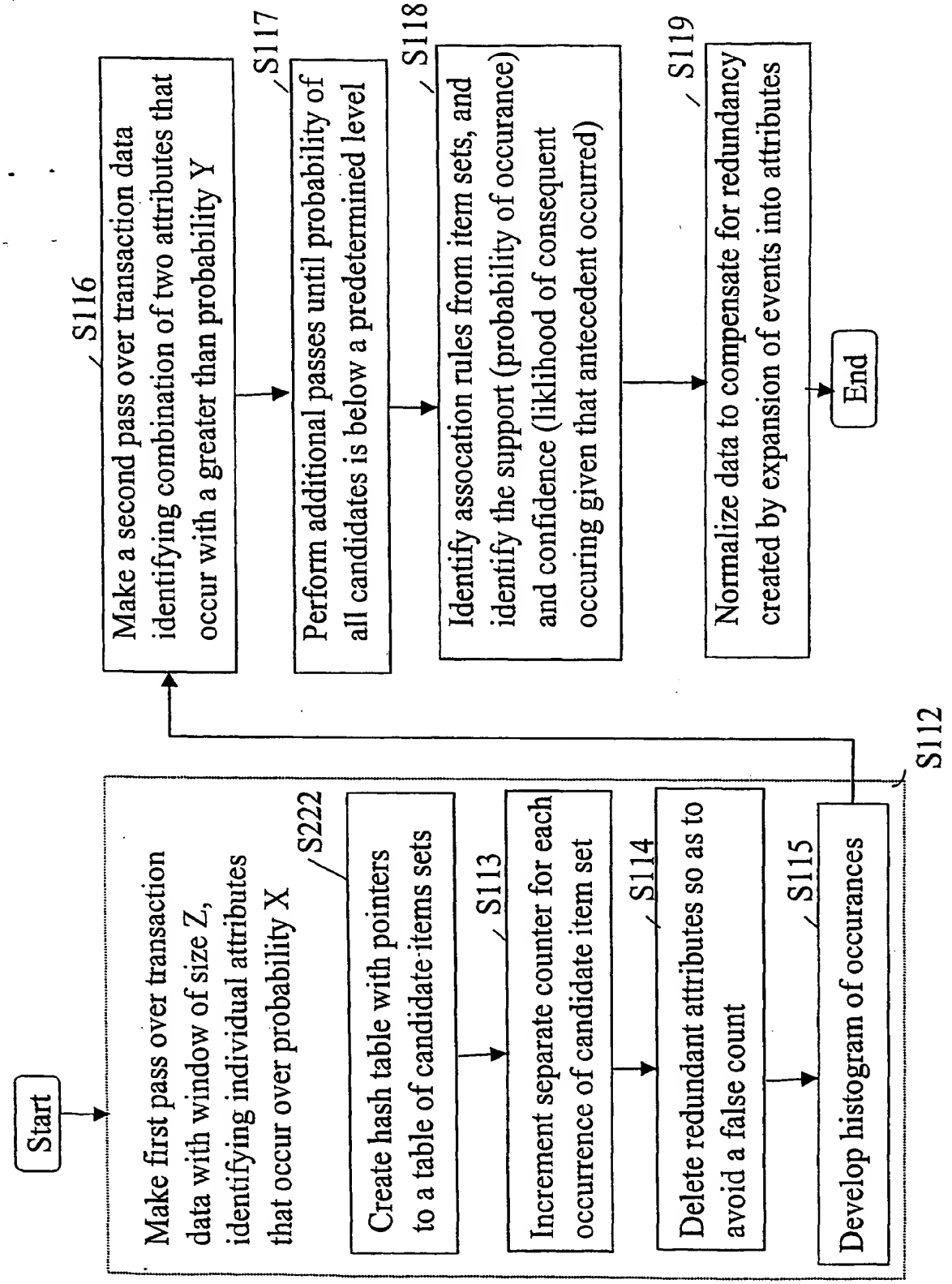

Start
Make first pass over transaction data with window of size Z, identifying individual attributes that occur over probability X
S222
Create hash table with pointers to a table of candidate items sets
S113
Increment separate counter for each occurrence of candidate item set
S114
Delete redundant attributes so as to avoid a false count
S115
Develop histogram of occurances
S112
S116
Make a second pass over transaction data identifying combination of two attributes that occur with a greater than probability Y
S117
Perform additional passes until probability of all candidates is below a predetermined level
S118
Identify assocation rules from item sets, and identify the support (probability of occurance) and confidence (liklihood of consequent occuring given that antecedent occurred)
S119
Normalize data to compensate for redundancy created by expansion of events into attributes
End

FIG. 13

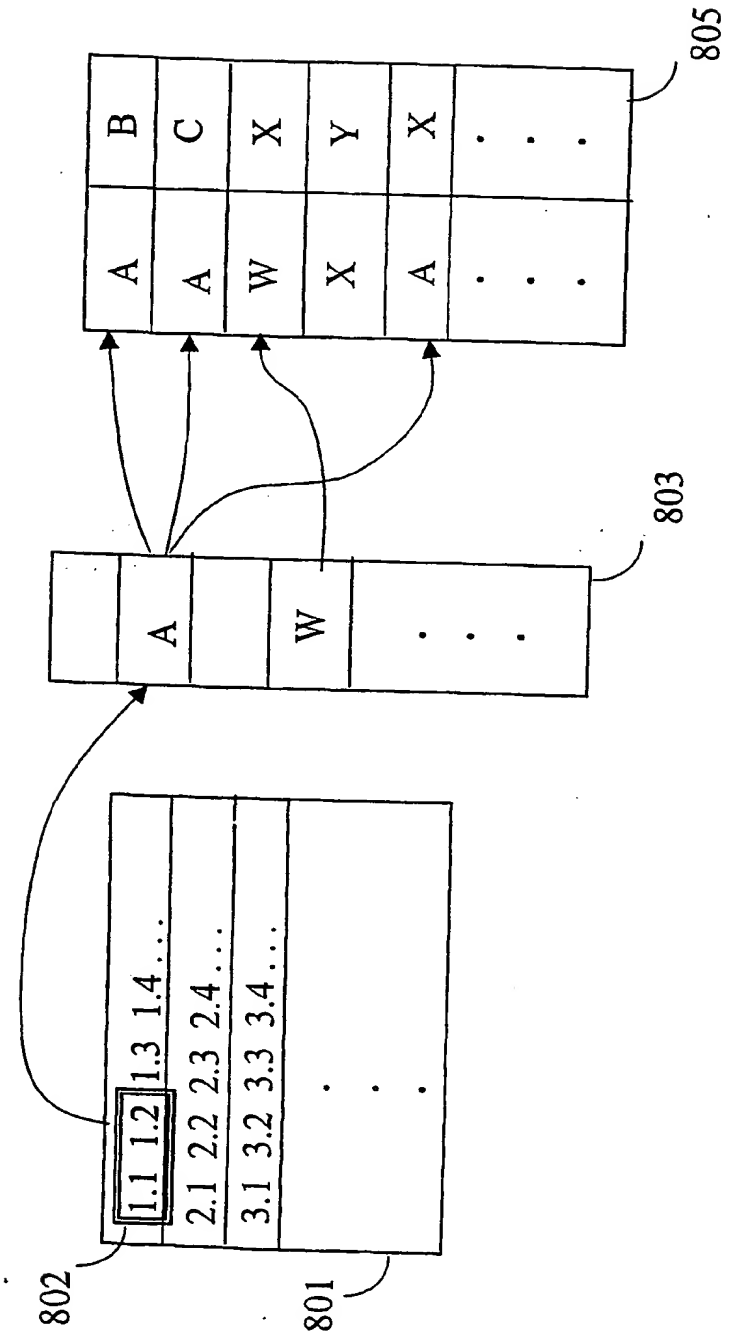


FIG. 14

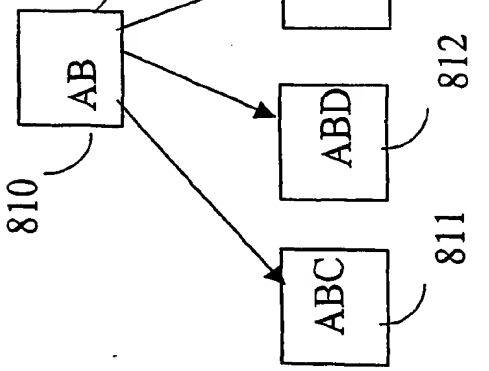


FIG. 15a

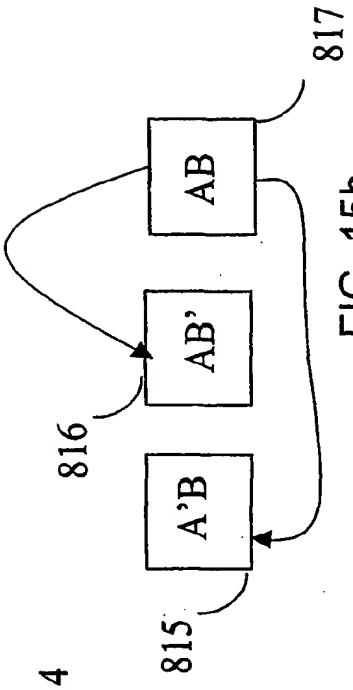


FIG. 15b

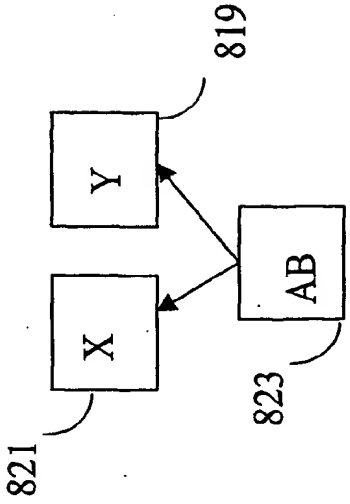


FIG. 15c

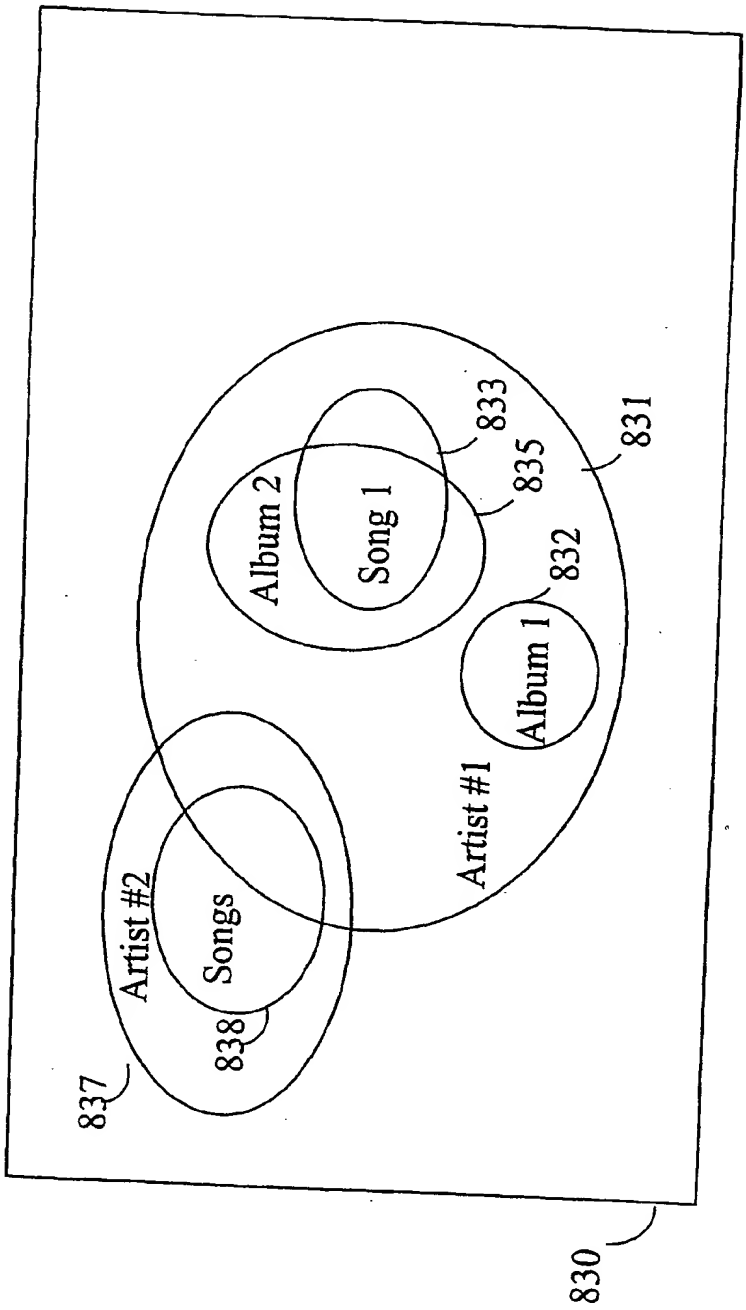

830
831
833
835
832
Album 2
Song 1
Album 1
Artist #1
837
838
Artist #2
Songs

FIG. 16

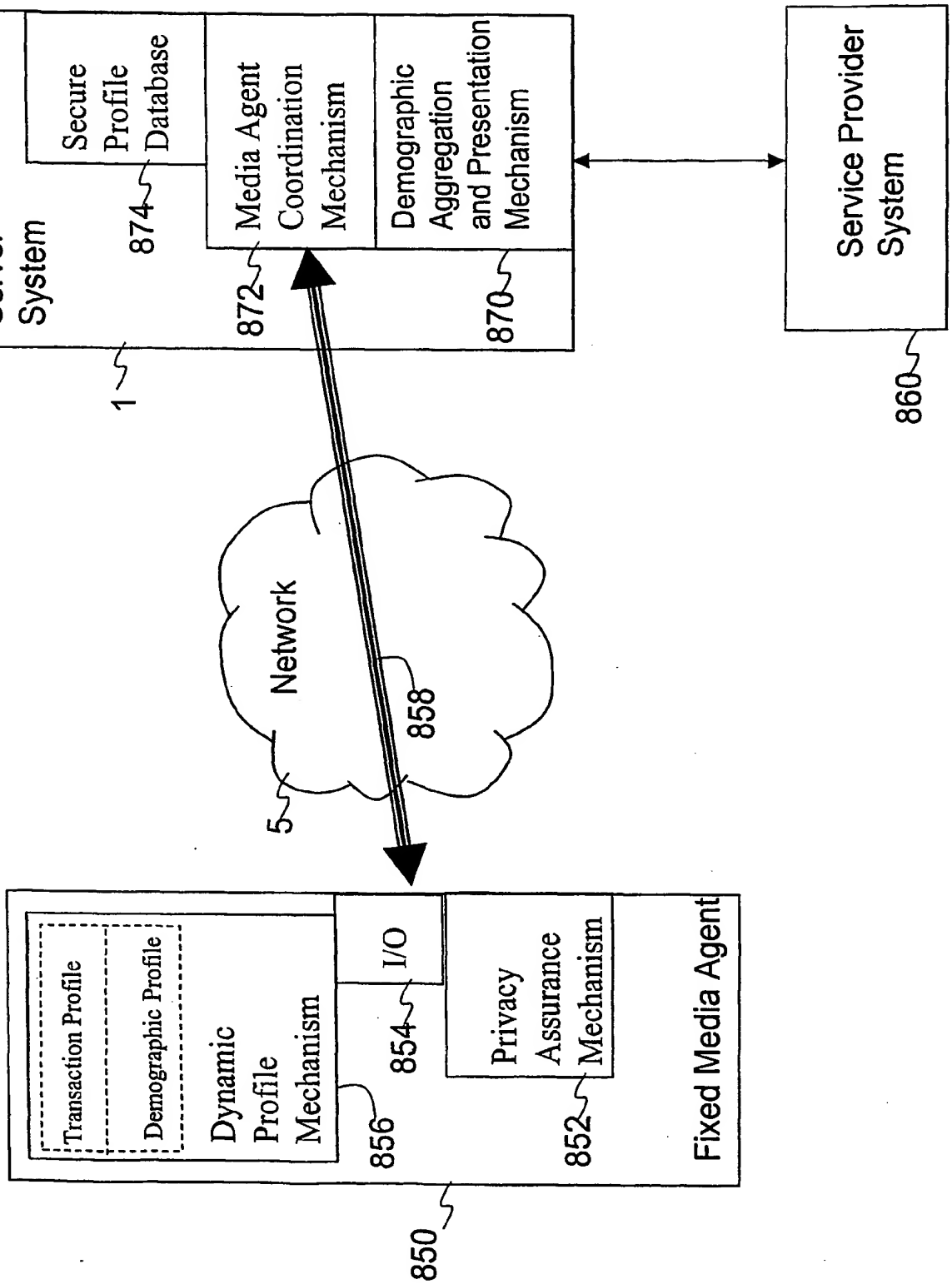

Server System
Secure Profile Database
874
Media Agent Coordination Mechanism
872
Demographic Aggregation and Presentation Mechanism
870
1
Service Provider System
860
Network
5
858
Transaction Profile
Demographic Profile
Dynamic Profile Mechanism
856
I/O
854
Privacy Assurance Mechanism
852
Fixed Media Agent
850

FIG. 17

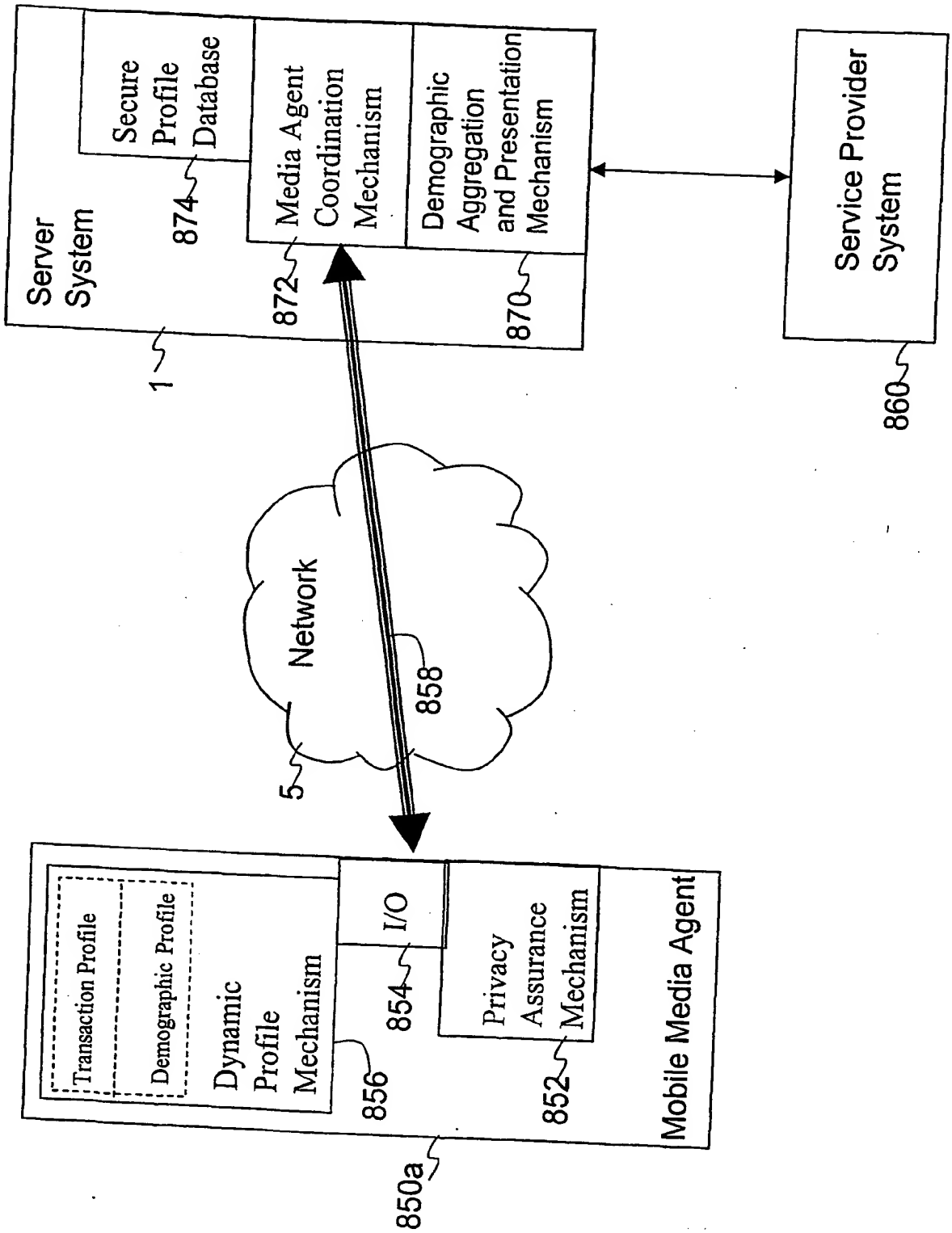

Server System
Secure Profile Database
874
Media Agent Coordination Mechanism
872
Demographic Aggregation and Presentation Mechanism
870
1
Service Provider System
860
Network
858
5
Transaction Profile
Demographic Profile
Dynamic Profile Mechanism
856
I/O
854
Privacy Assurance Mechanism
852
Mobile Media Agent
850a

FIG. 18

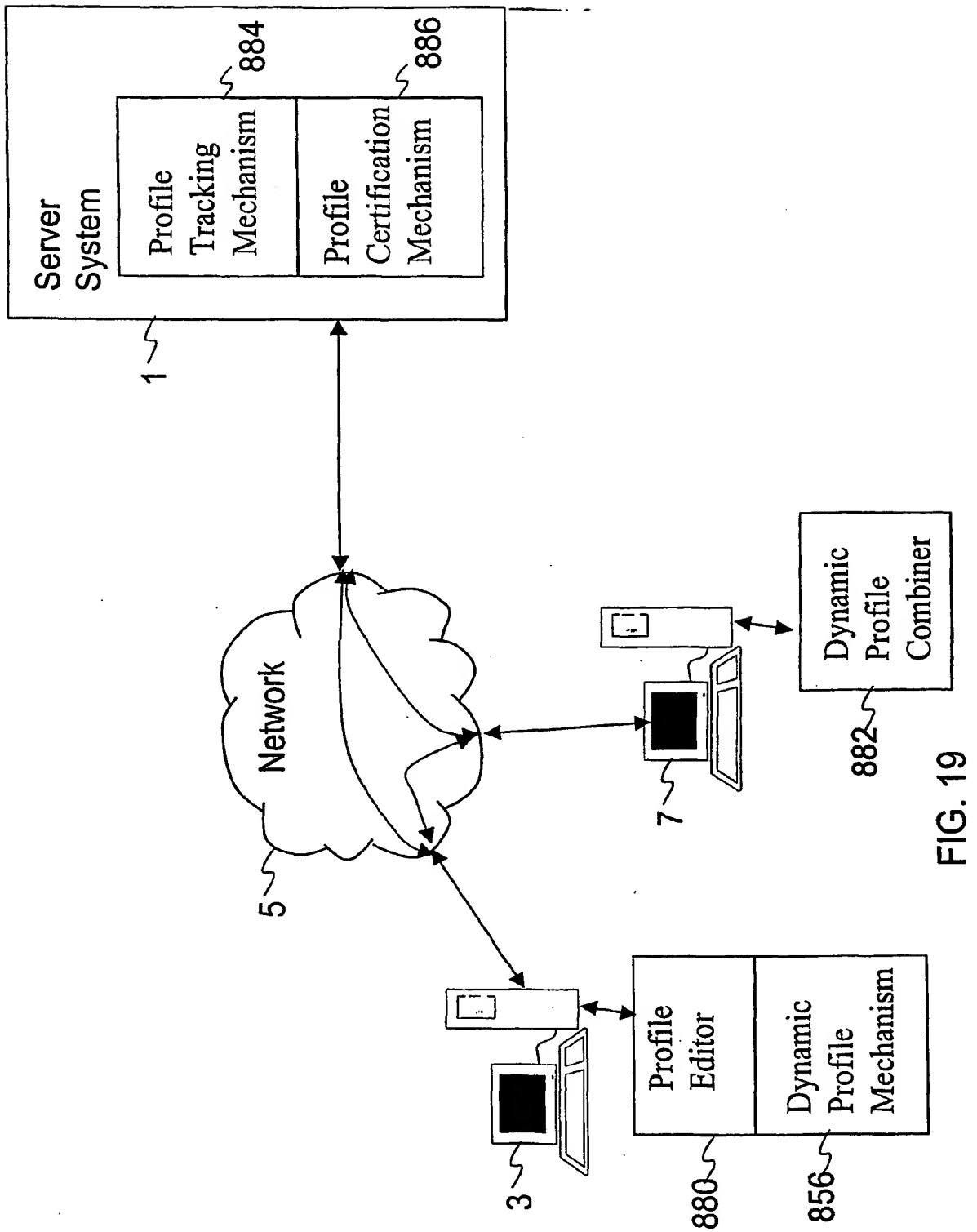


FIG. 19

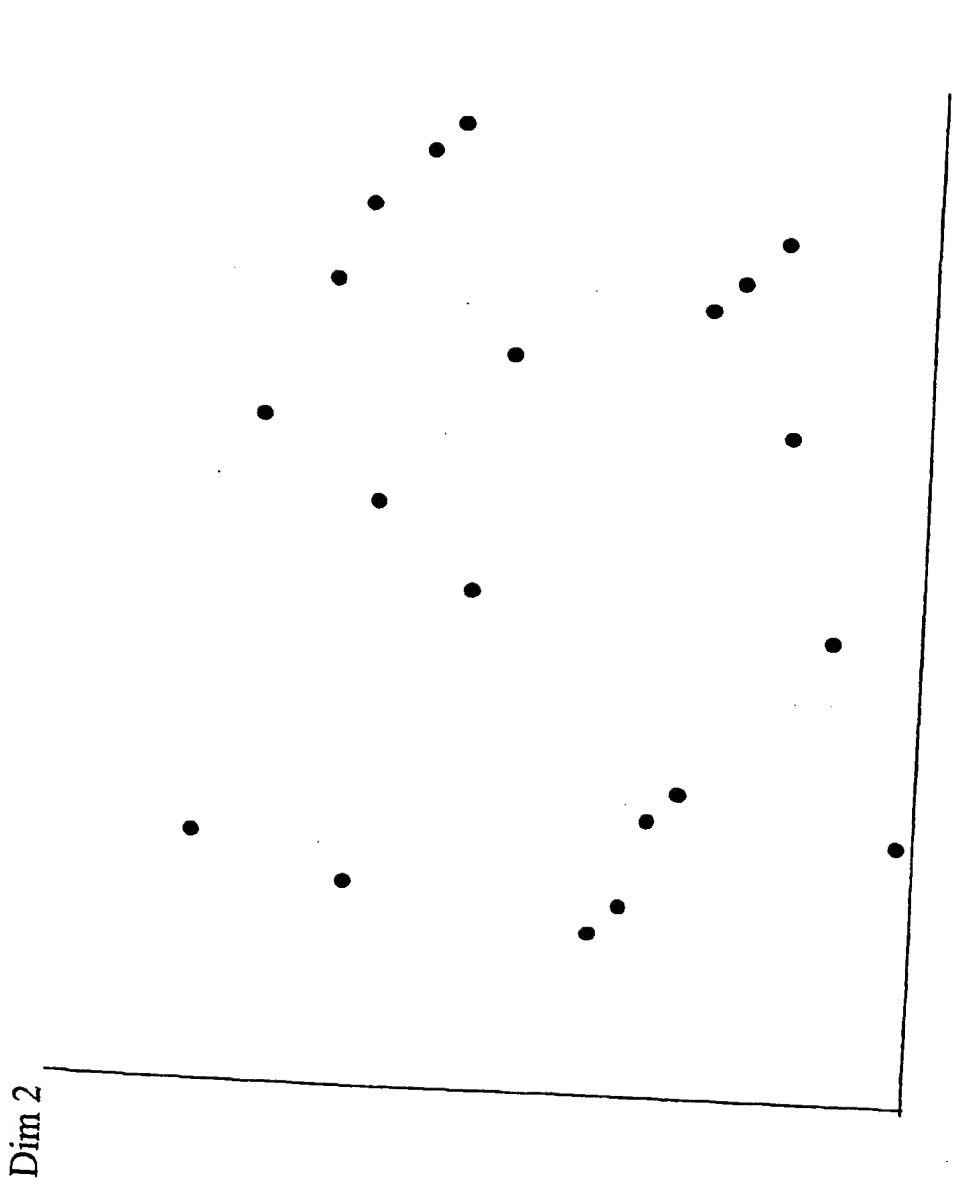


FIG. 20

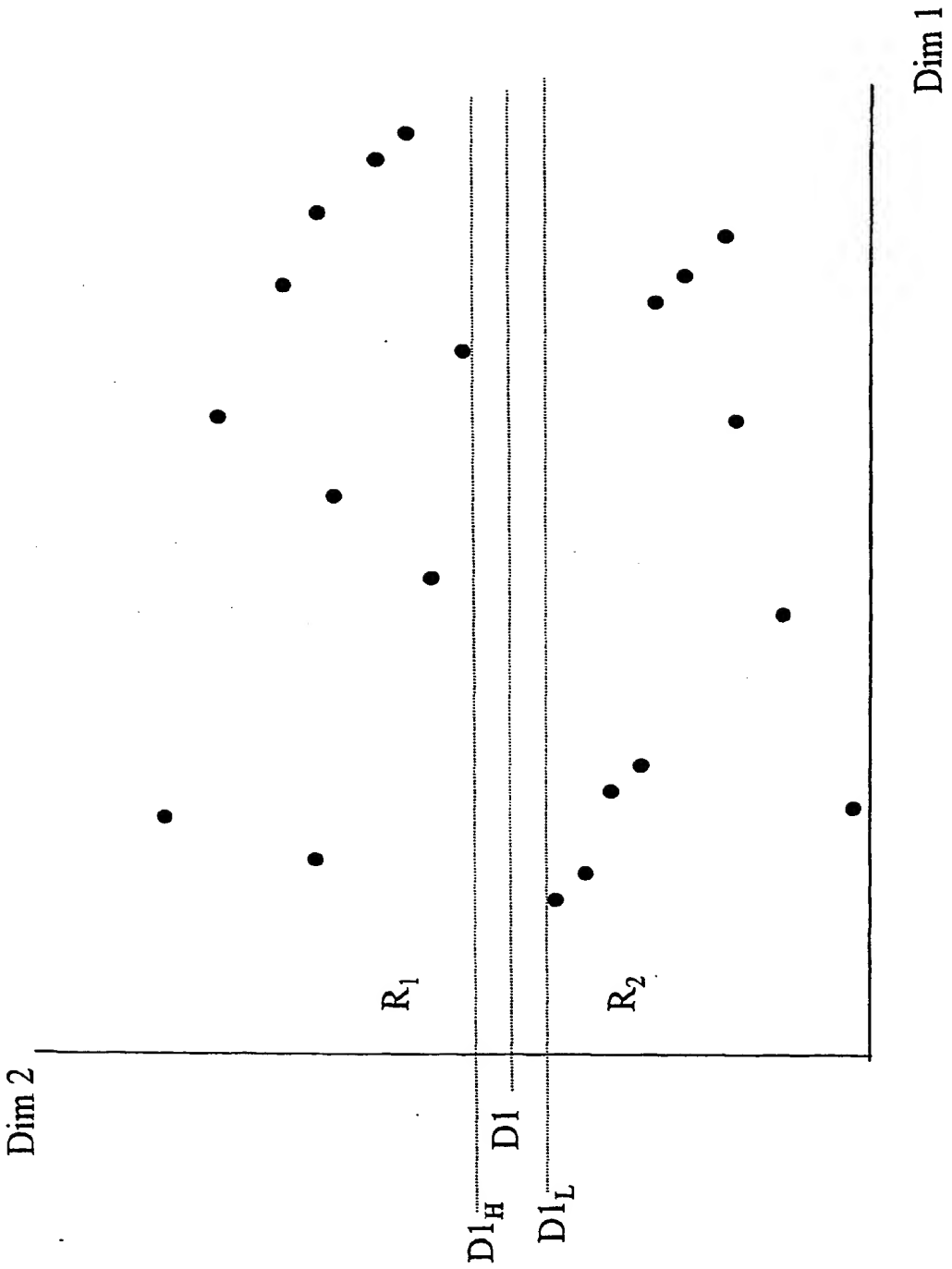


FIG. 21

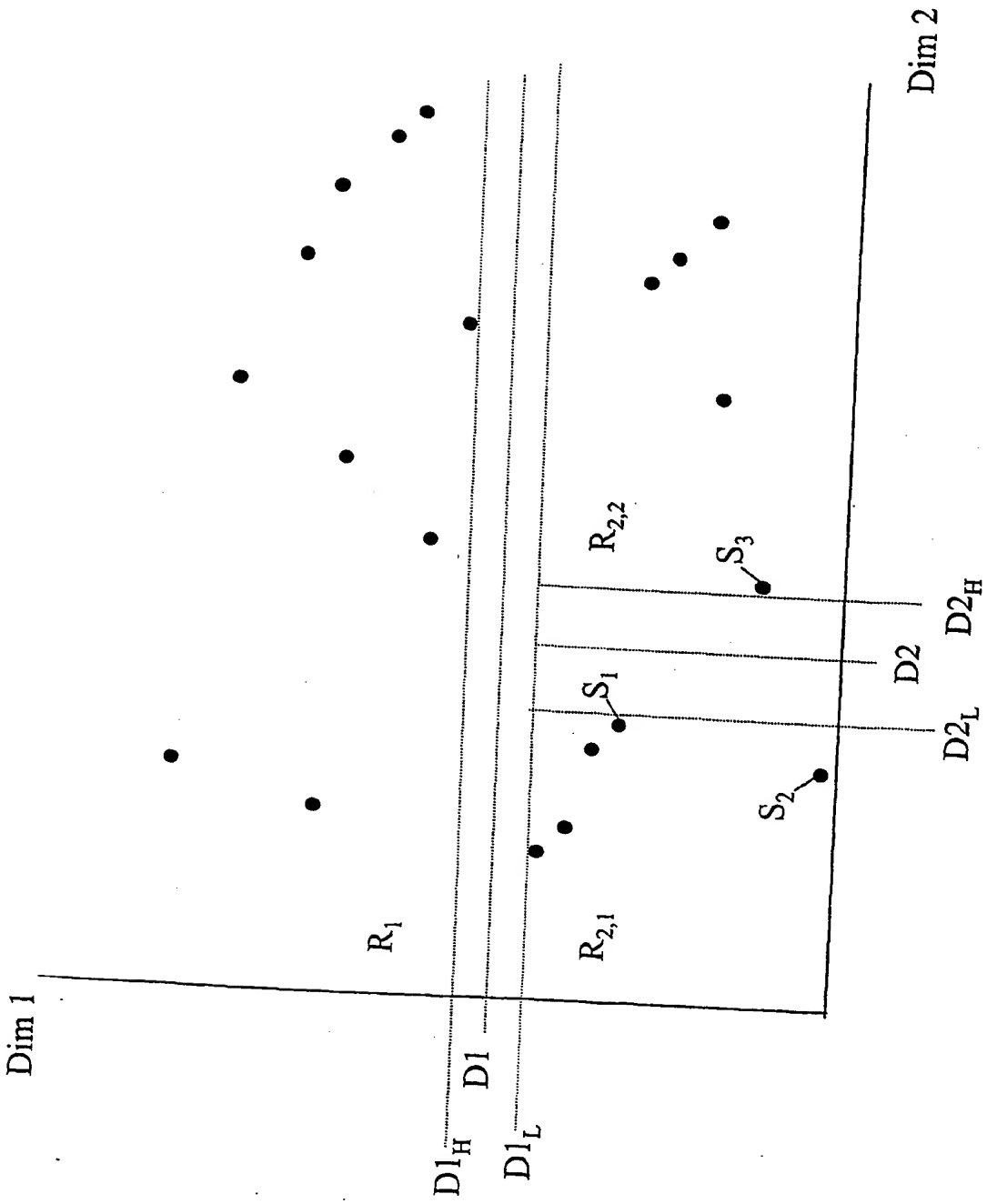


FIG. 22

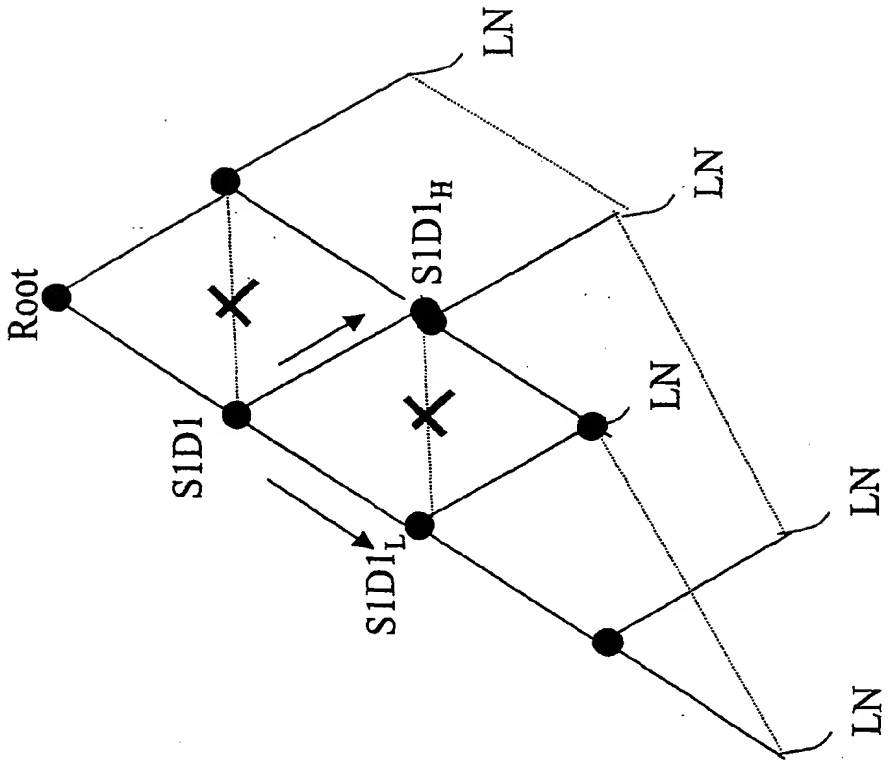


FIG. 23

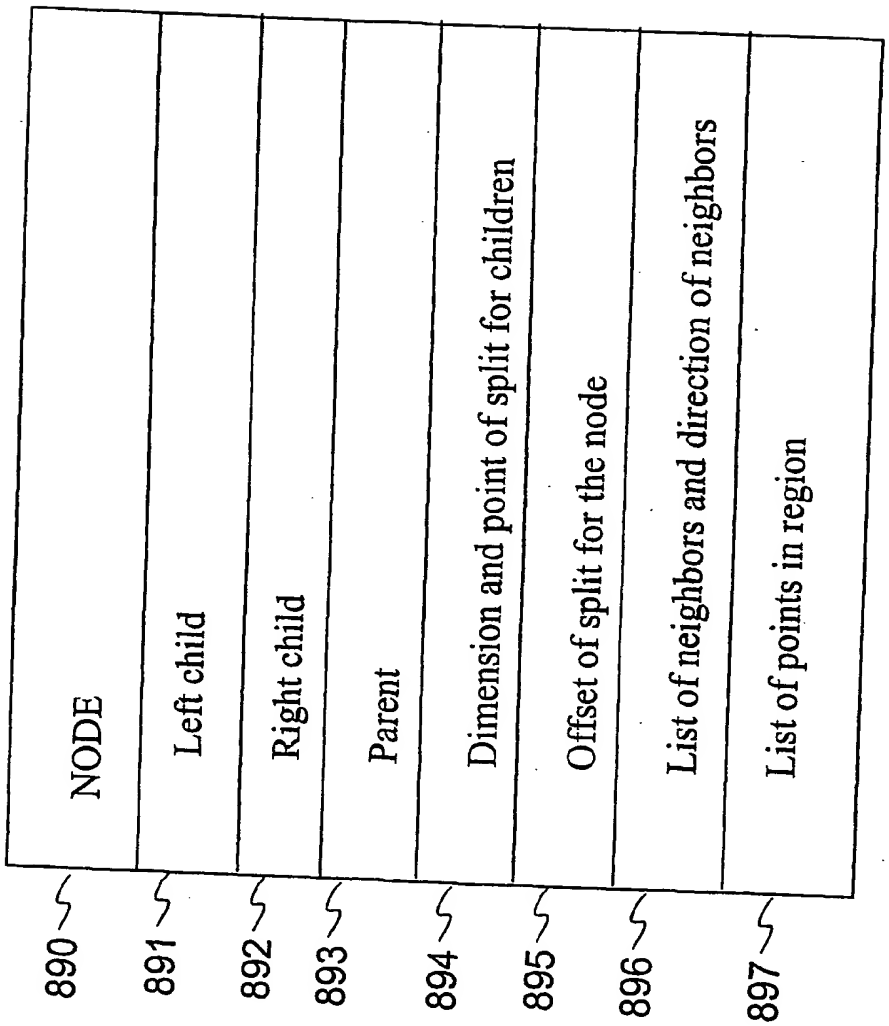

NODE
Left child
Right child
Parent
Dimension and point of split for children
Offset of split for the node
List of neighbors and direction of neighbors
List of points in region
890
891
892
893
894
895
896
897

FIG. 24

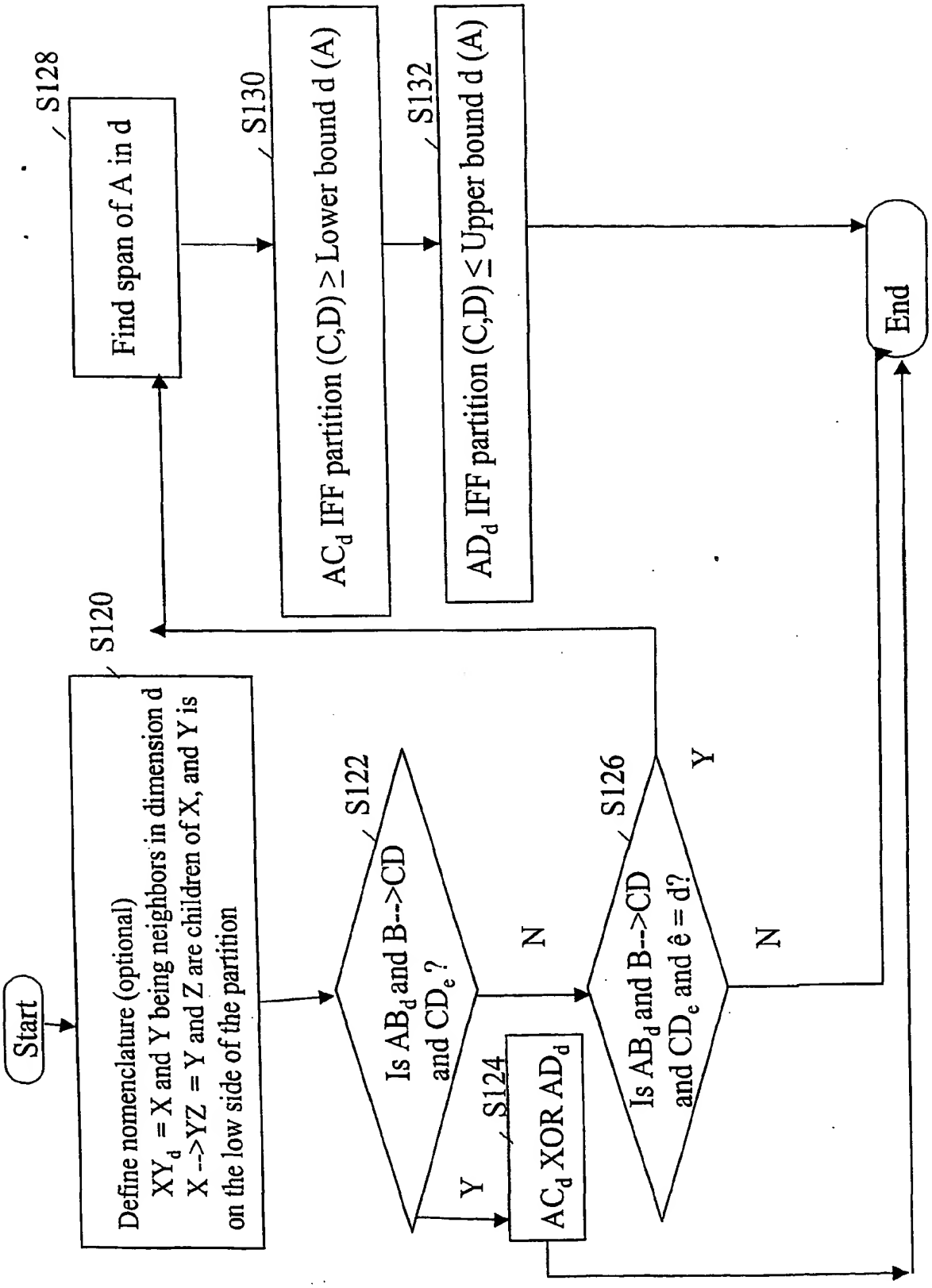

Start
S120
Define nomenclature (optional)
XY_d = X and Y being neighbors in dimension d
X -->YZ = Y and Z are children of X, and Y is on the low side of the partition
S122
Is AB_d and B-->CD and CD_e ?
Y
S124
AC_d XOR AD_d
N
S126
Is AB_d and B-->CD and CD_e and ê = d?
Y
N
S128
Find span of A in d
S130
AC_d IFF partition (C,D) ≥ Lower bound d (A)
S132
AD_d IFF partition (C,D) ≤ Upper bound d (A)
End

FIG. 25

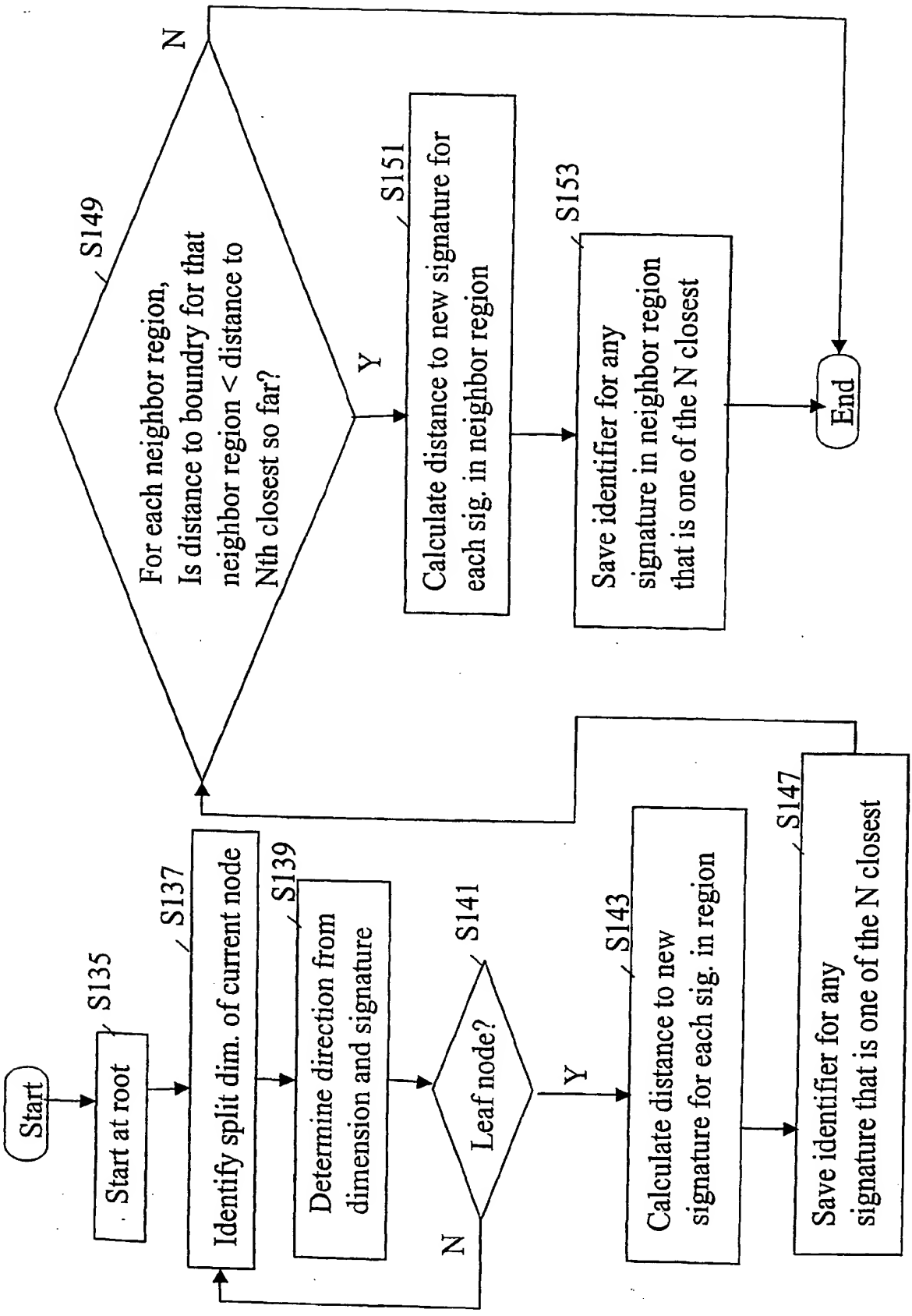

Start
Start at root
S135
Identify split dim. of current node
S137
Determine direction from dimension and signature
S139
Leaf node?
S141
N
Y
Calculate distance to new signature for each sig. in region
S143
Save identifier for any signature that is one of the N closest
S147
For each neighbor region, Is distance to boundry for that neighbor region < distance to Nth closest so far?
S149
N
Y
Calculate distance to new signature for each sig. in neighbor region
S151
Save identifier for any signature in neighbor region that is one of the N closest
S153
End

FIG. 26

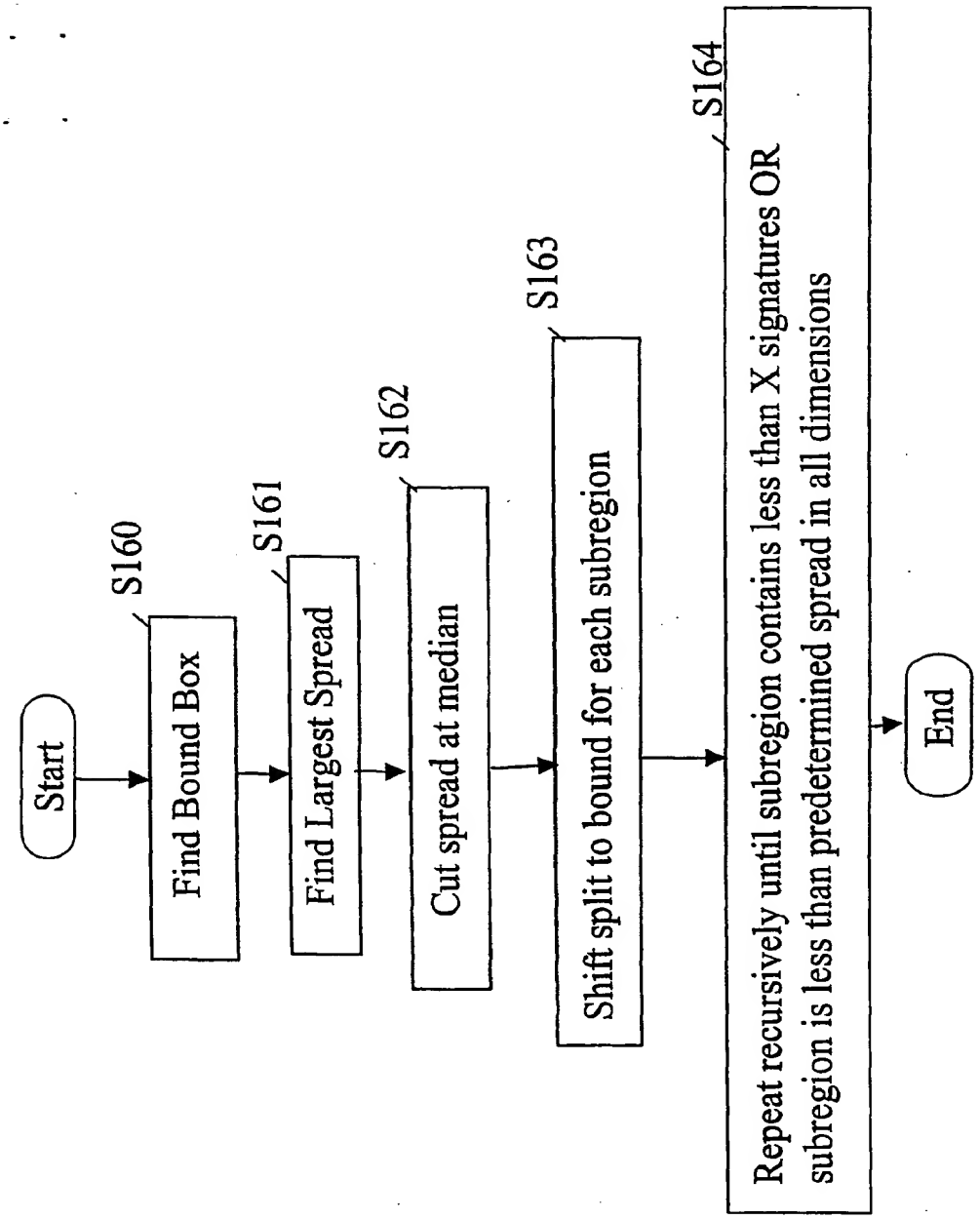


FIG. 27

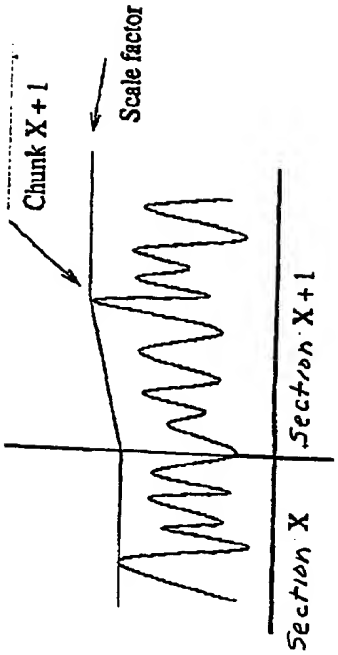


Fig. 28

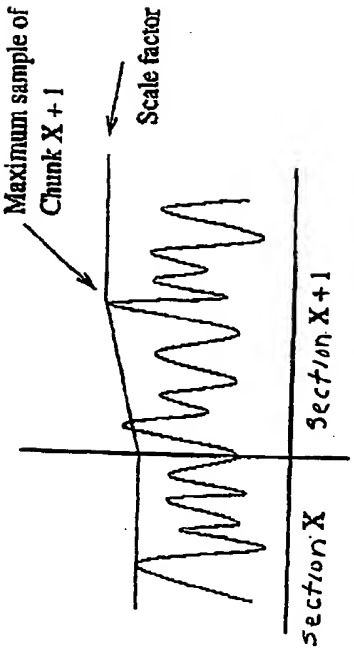
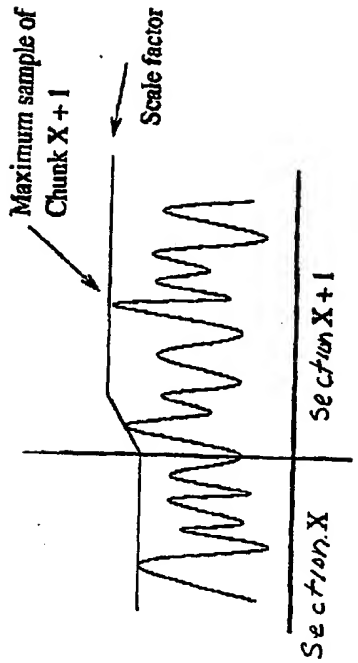


Fig. 29



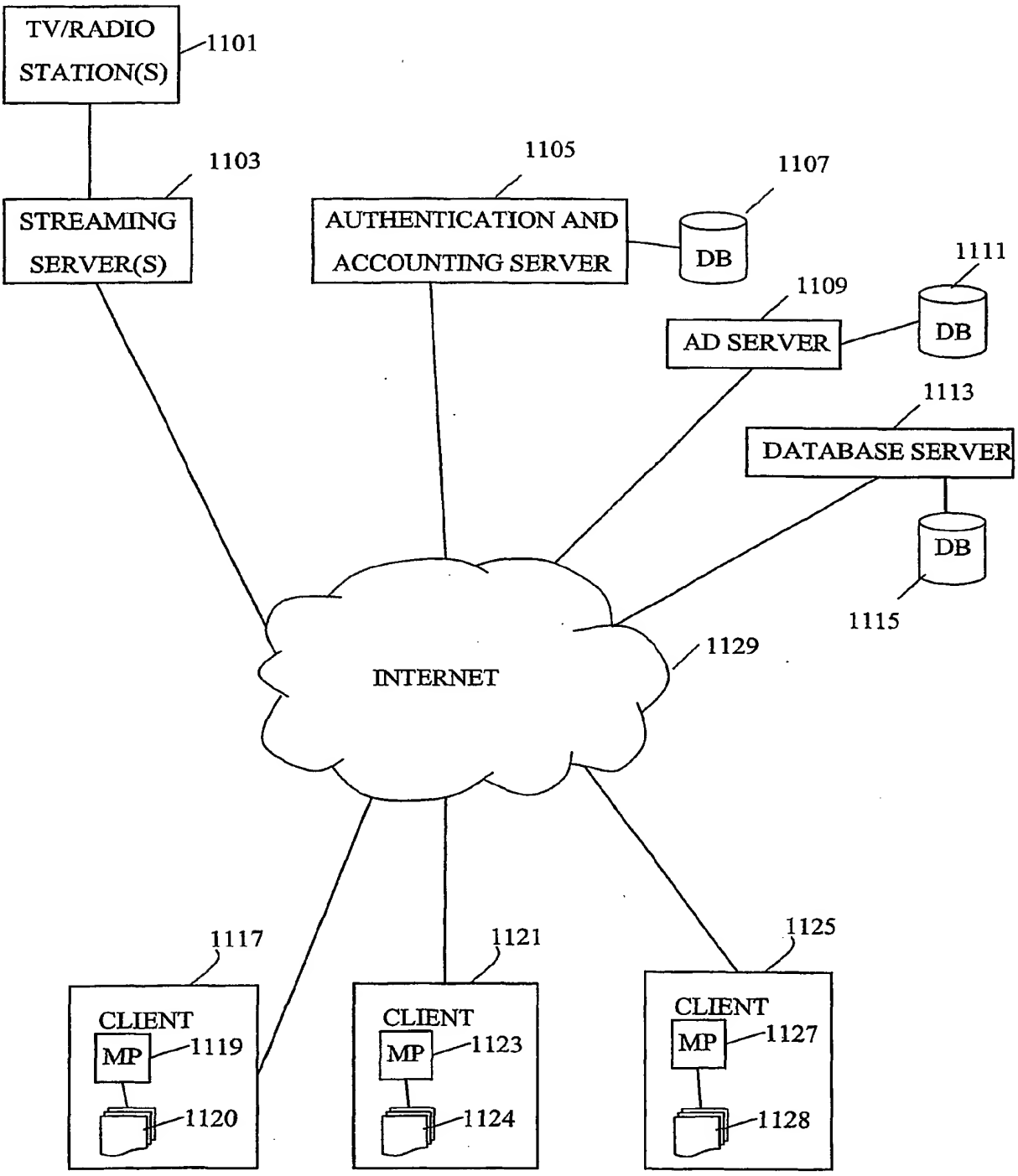

TV/RADIO STATION(S)
1101
1103
STREAMING SERVER(S)
1105
AUTHENTICATION AND ACCOUNTING SERVER
1107
DB
1111
1109
AD SERVER
DB
1113
DATABASE SERVER
DB
1115
INTERNET
1129
1117
CLIENT
MP
1119
1120
1121
CLIENT
MP
1123
1124
1125
CLIENT
MP
1127
1128

FIG. 31

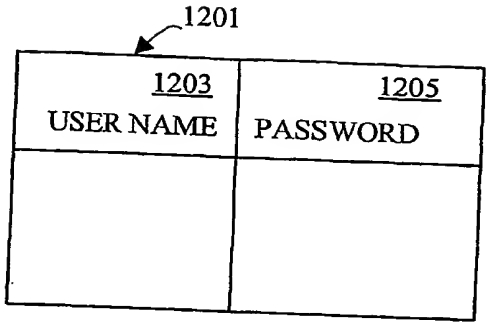


FIG. 32

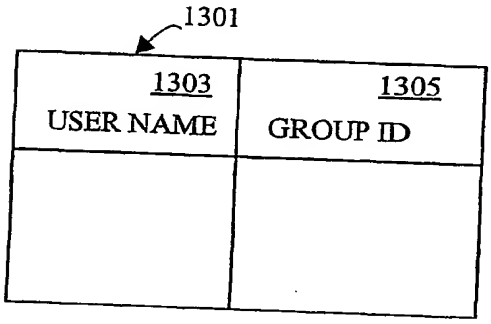


FIG. 33

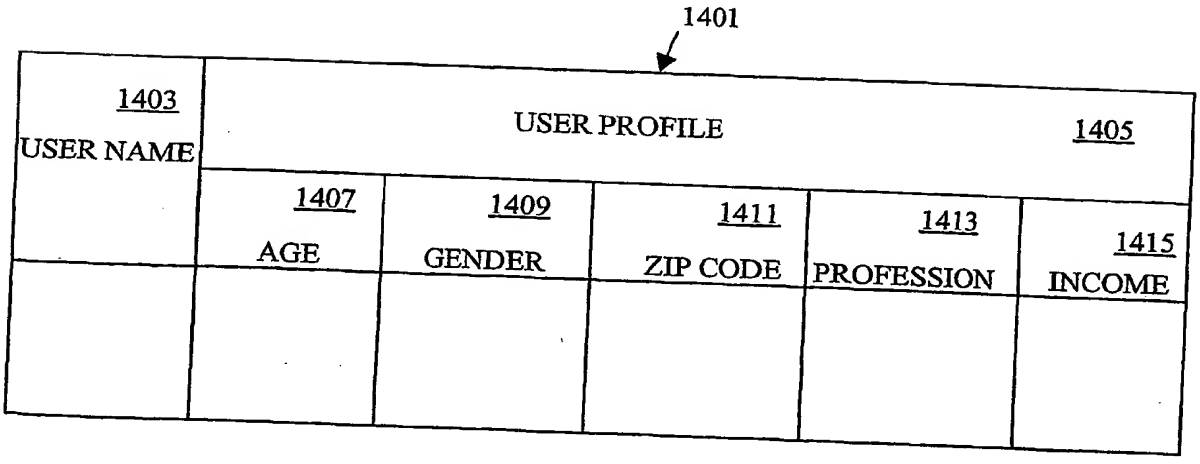


FIG. 34

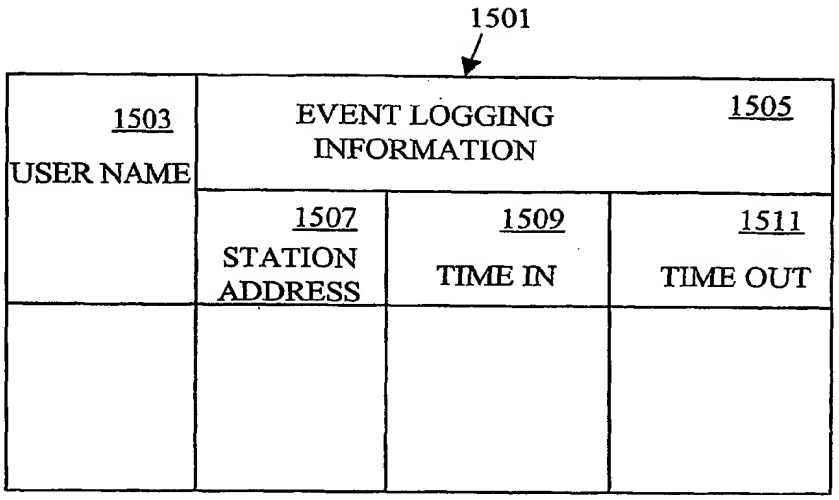


FIG. 35

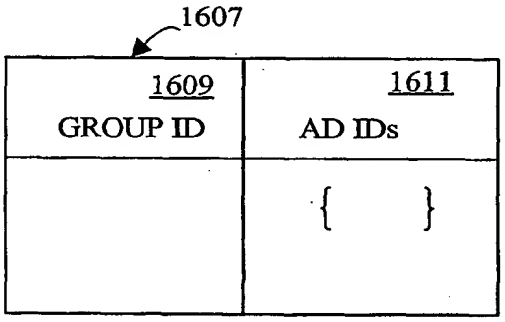


FIG. 36

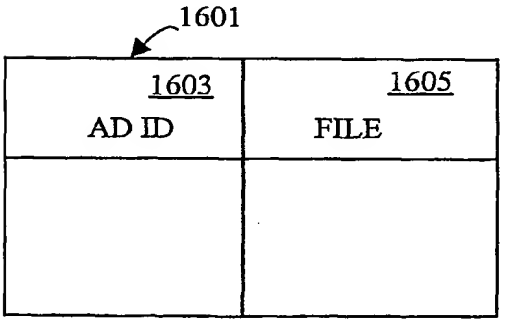


FIG. 37

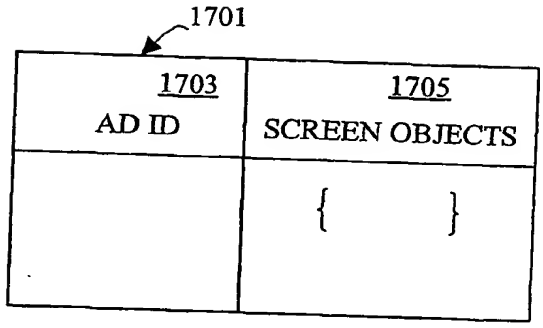


FIG. 38

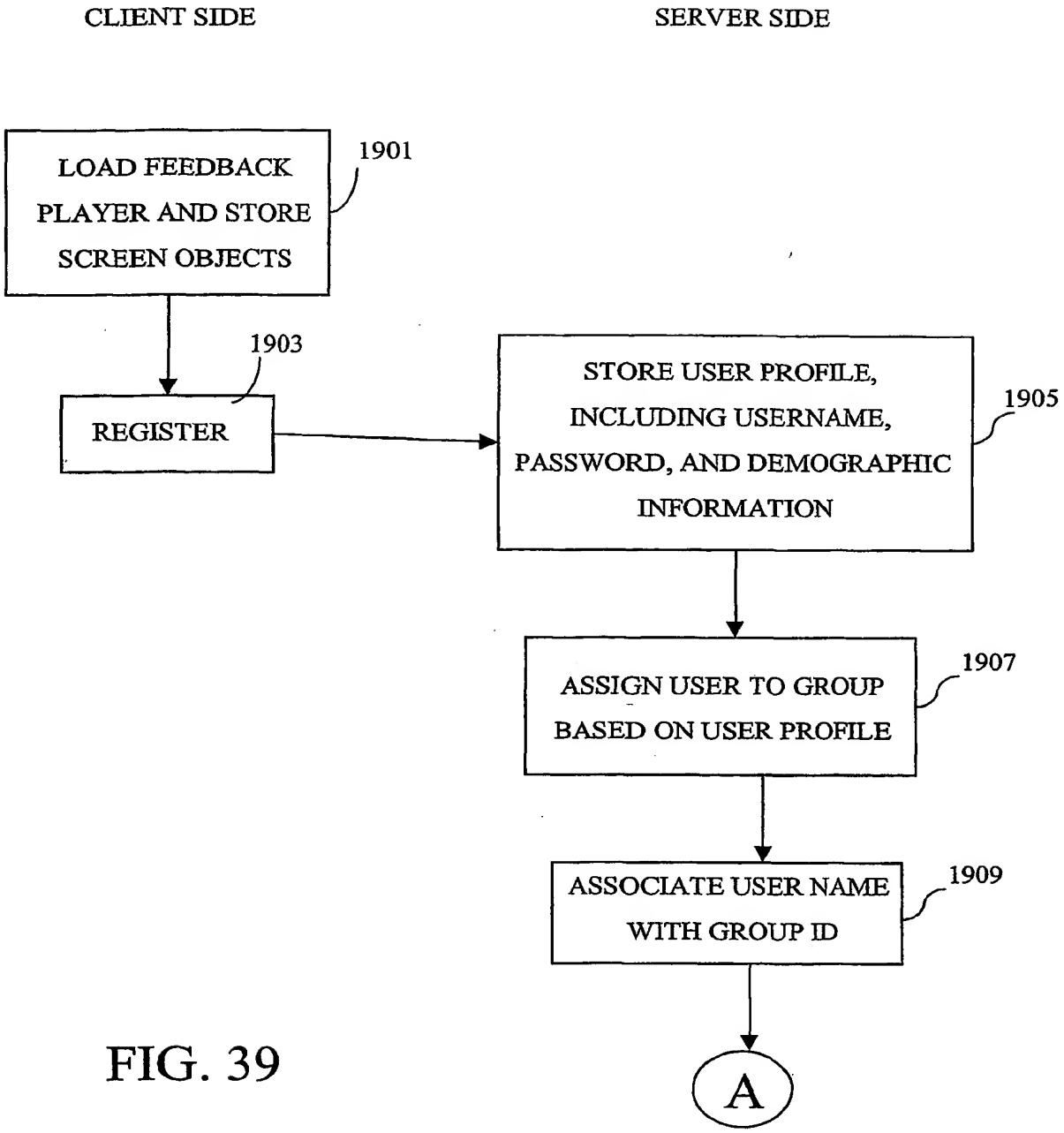

CLIENT SIDE
SERVER SIDE
LOAD FEEDBACK PLAYER AND STORE SCREEN OBJECTS
1901
1903
REGISTER
STORE USER PROFILE, INCLUDING USERNAME, PASSWORD, AND DEMOGRAPHIC INFORMATION
1905
ASSIGN USER TO GROUP BASED ON USER PROFILE
1907
ASSOCIATE USER NAME WITH GROUP ID
1909
A

FIG. 39

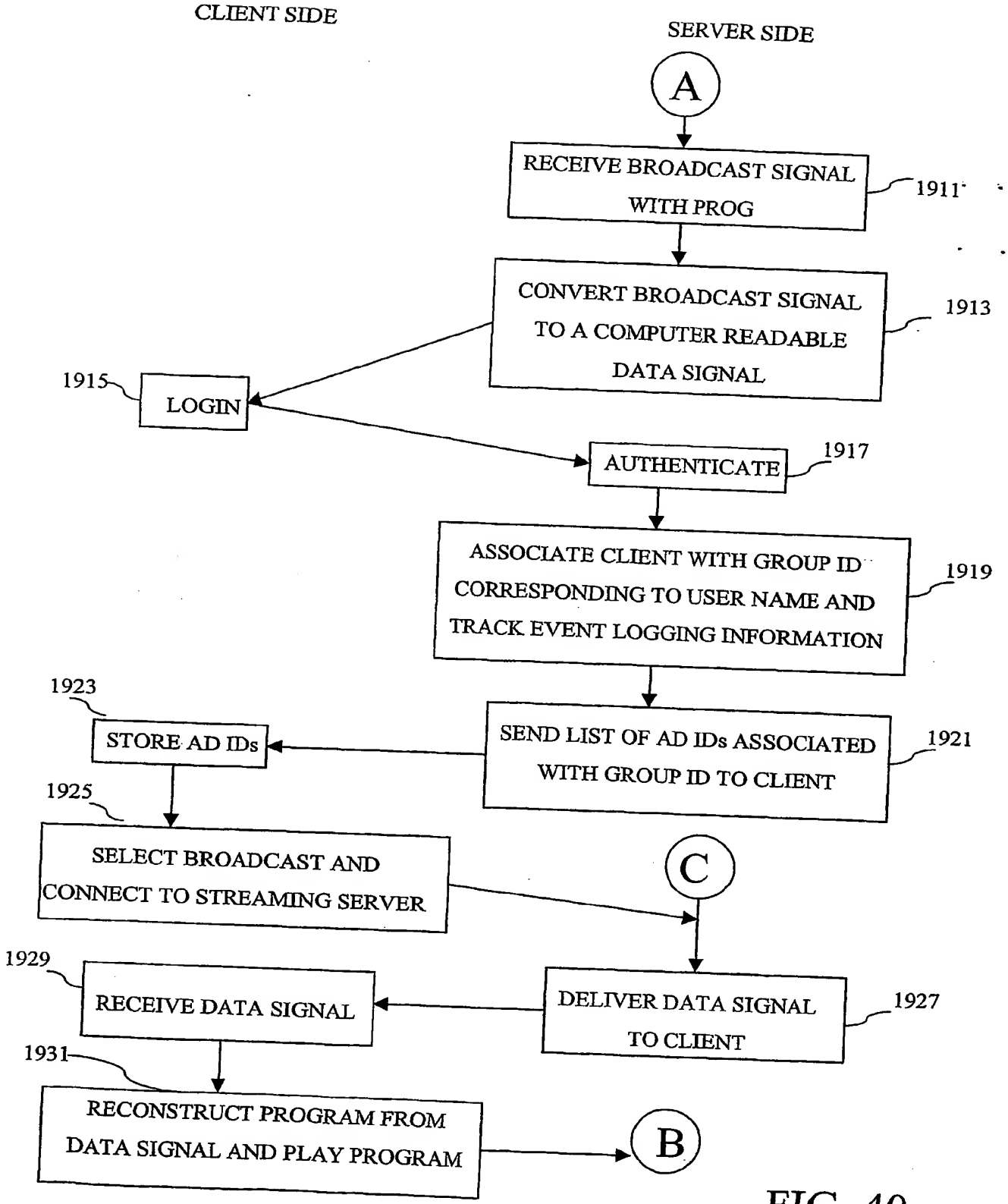

CLIENT SIDE
SERVER SIDE
A
RECEIVE BROADCAST SIGNAL WITH PROG
1911
CONVERT BROADCAST SIGNAL TO A COMPUTER READABLE DATA SIGNAL
1913
1915
LOGIN
AUTHENTICATE
1917
ASSOCIATE CLIENT WITH GROUP ID CORRESPONDING TO USER NAME AND TRACK EVENT LOGGING INFORMATION
1919
1923
STORE AD IDs
SEND LIST OF AD IDs ASSOCIATED WITH GROUP ID TO CLIENT
1921
1925
SELECT BROADCAST AND CONNECT TO STREAMING SERVER
C
1929
RECEIVE DATA SIGNAL
DELIVER DATA SIGNAL TO CLIENT
1927
1931
RECONSTRUCT PROGRAM FROM DATA SIGNAL AND PLAY PROGRAM
B

FIG. 40

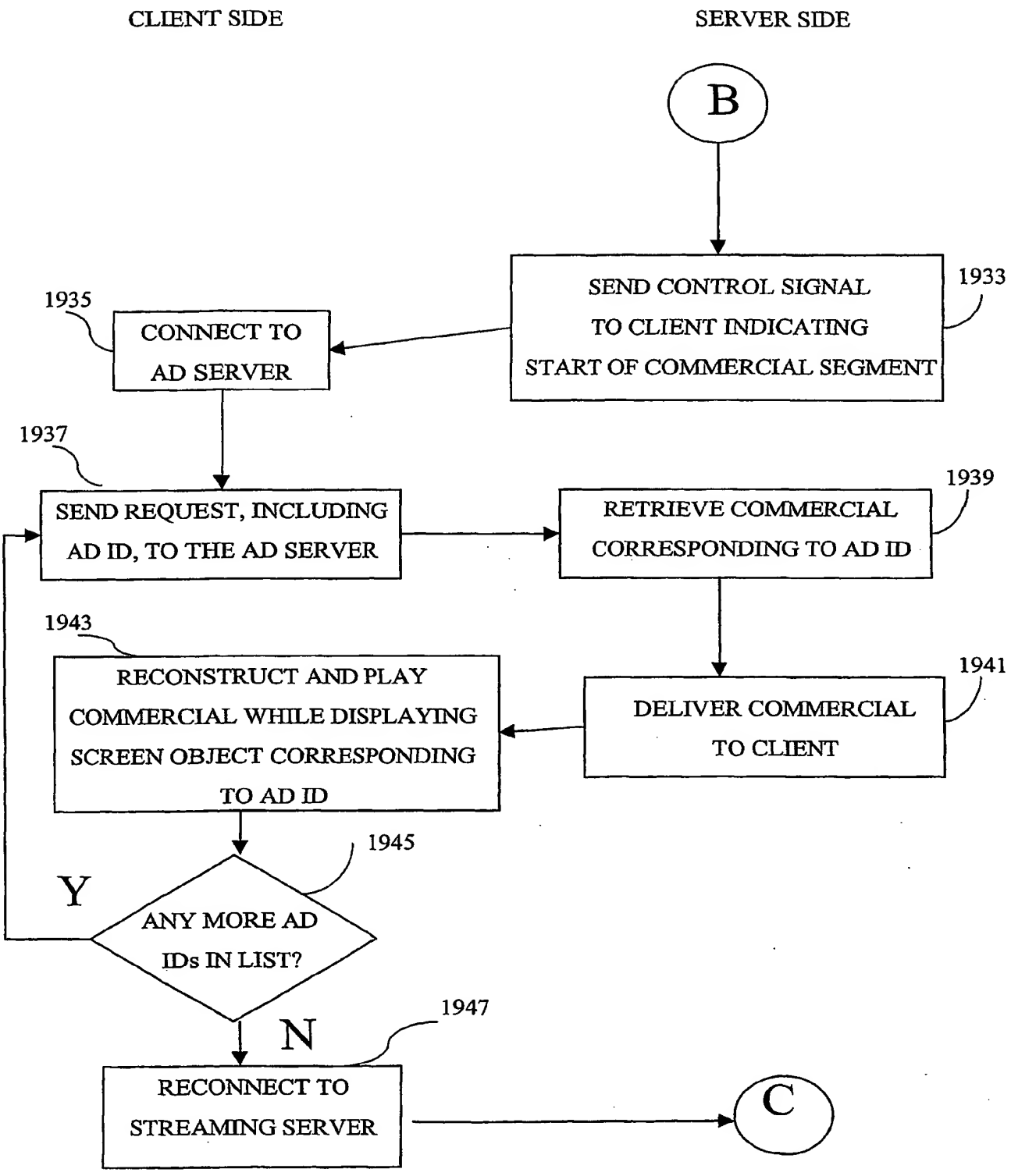

CLIENT SIDE
SERVER SIDE
B
SEND CONTROL SIGNAL TO CLIENT INDICATING START OF COMMERCIAL SEGMENT
1933
1935
CONNECT TO AD SERVER
1937
SEND REQUEST, INCLUDING AD ID, TO THE AD SERVER
1939
RETRIEVE COMMERCIAL CORRESPONDING TO AD ID
1941
DELIVER COMMERCIAL TO CLIENT
1943
RECONSTRUCT AND PLAY COMMERCIAL WHILE DISPLAYING SCREEN OBJECT CORRESPONDING TO AD ID
1945
Y
ANY MORE AD IDs IN LIST?
N
1947
RECONNECT TO STREAMING SERVER
C

FIG. 41

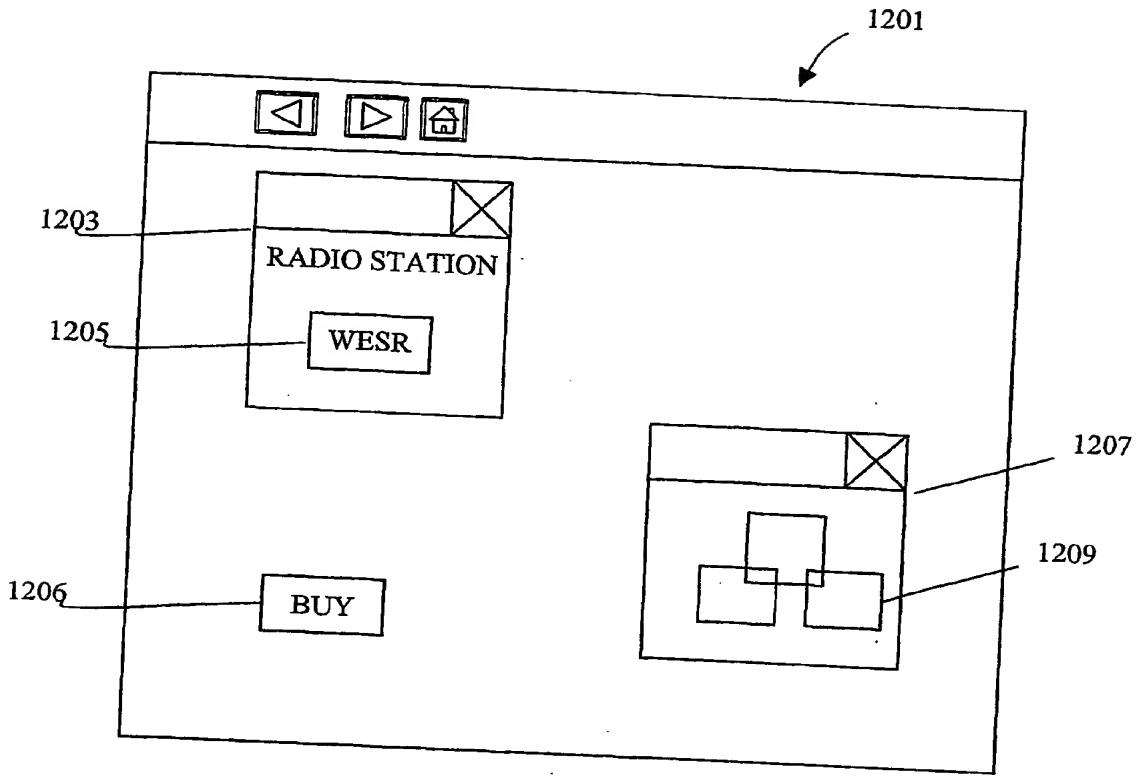
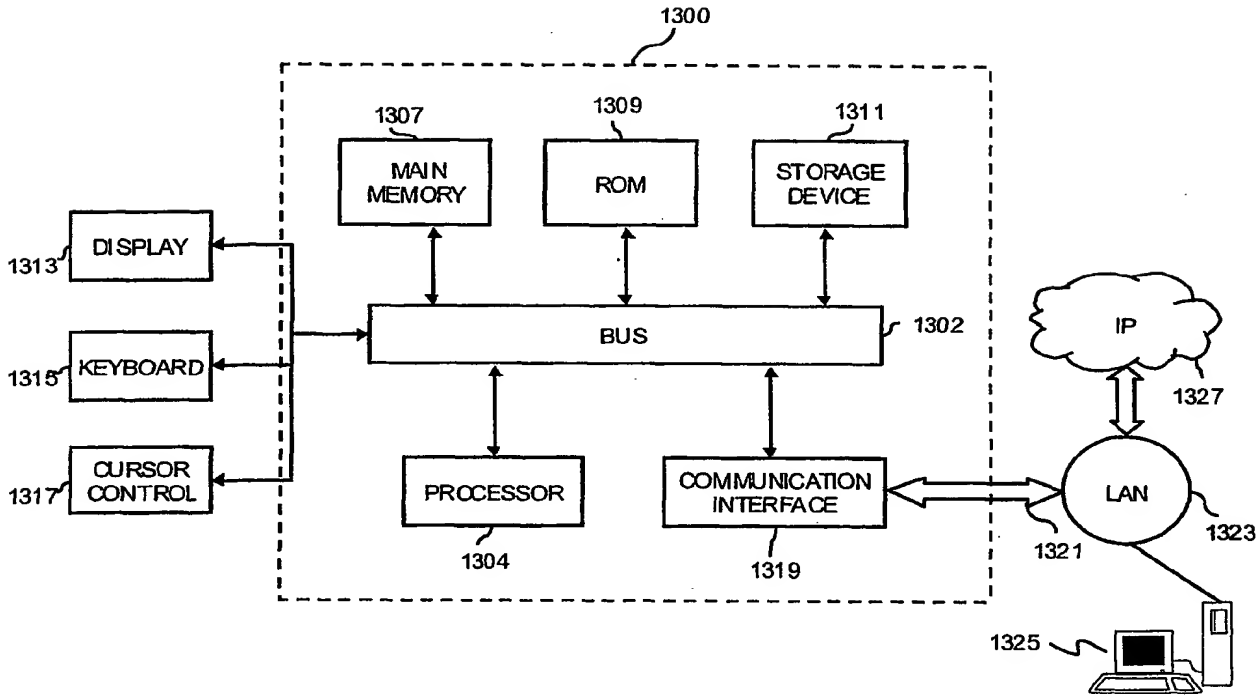


FIG. 42

Fig. 43


1300
1307
MAIN MEMORY
1309
ROM
1311
STORAGE DEVICE
DISPLAY
1313
KEYBOARD
1315
CURSOR CONTROL
1317
BUS
1302
PROCESSOR
1304
COMMUNICATION INTERFACE
1319
IP
1327
LAN
1323
1321
1325

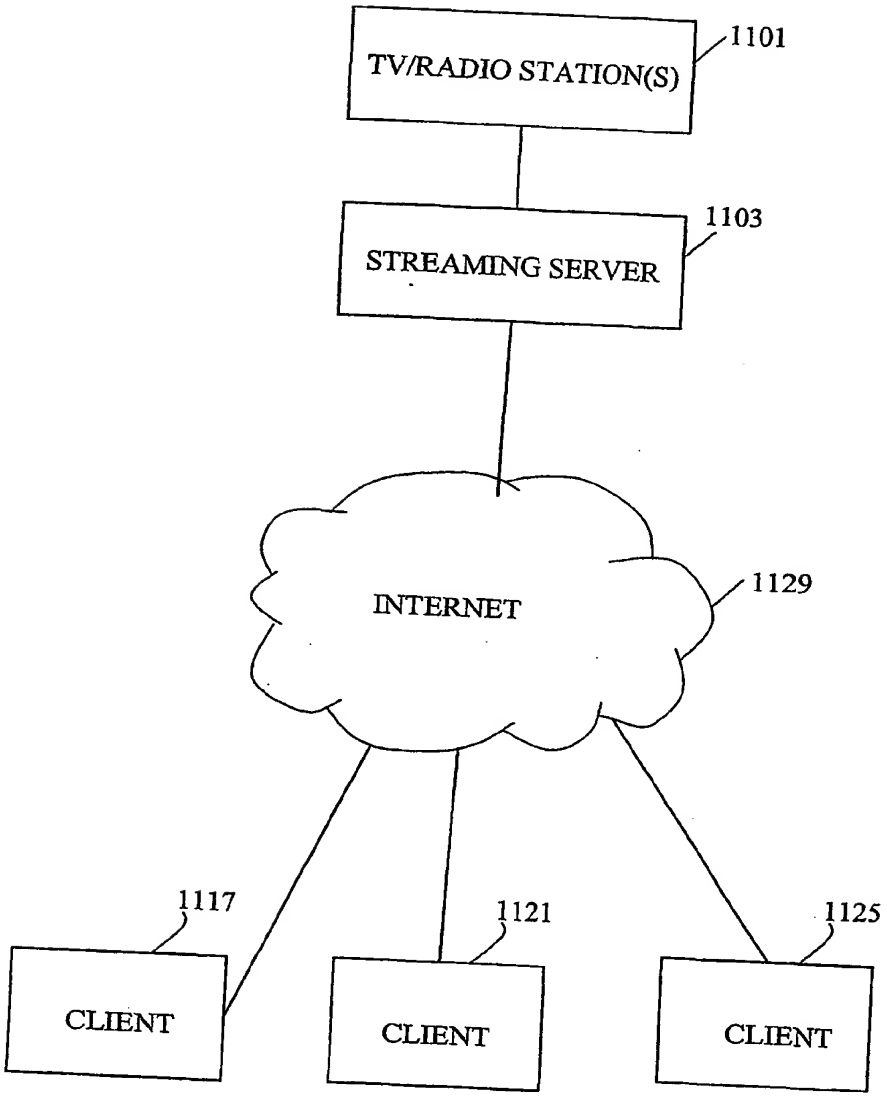


FIG. 44

(PRIOR ART)

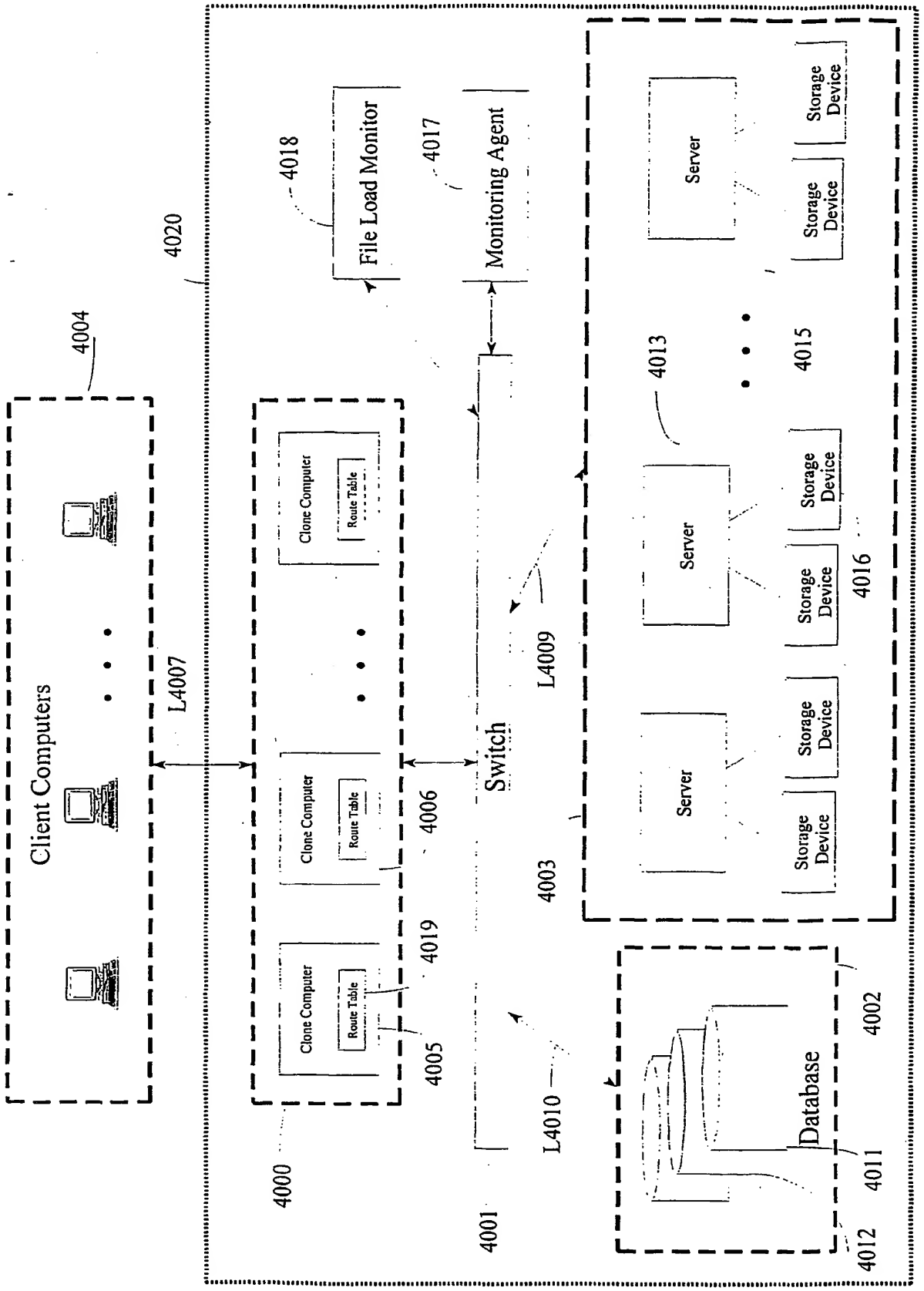


Fig. 45

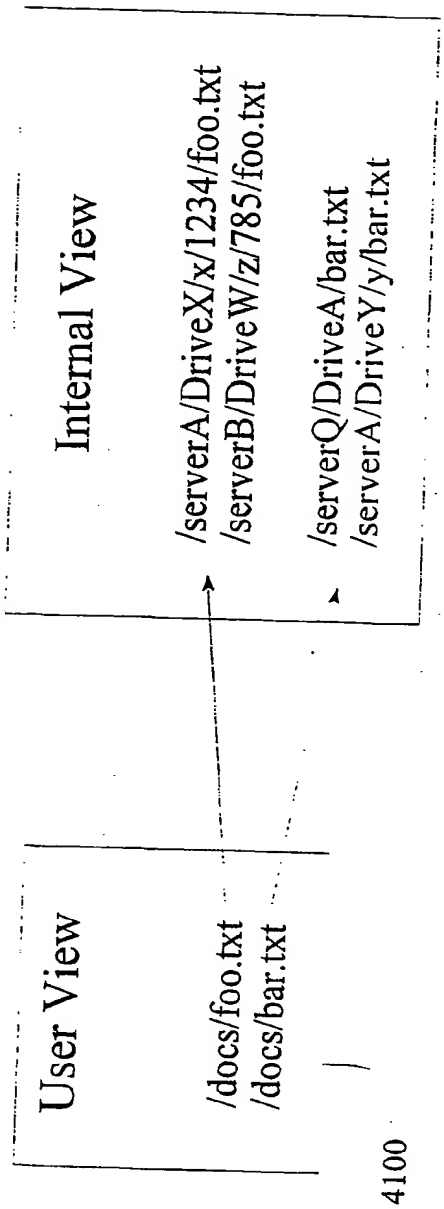

User View
/docs/foo.txt
/docs/bar.txt
4100
Internal View
/serverA/DriveX/x/1234/foo.txt
/serverB/DriveW/z/785/foo.txt
/serverQ/DriveA/bar.txt
/serverA/DriveY/y/bar.txt
4101

Fig. 46

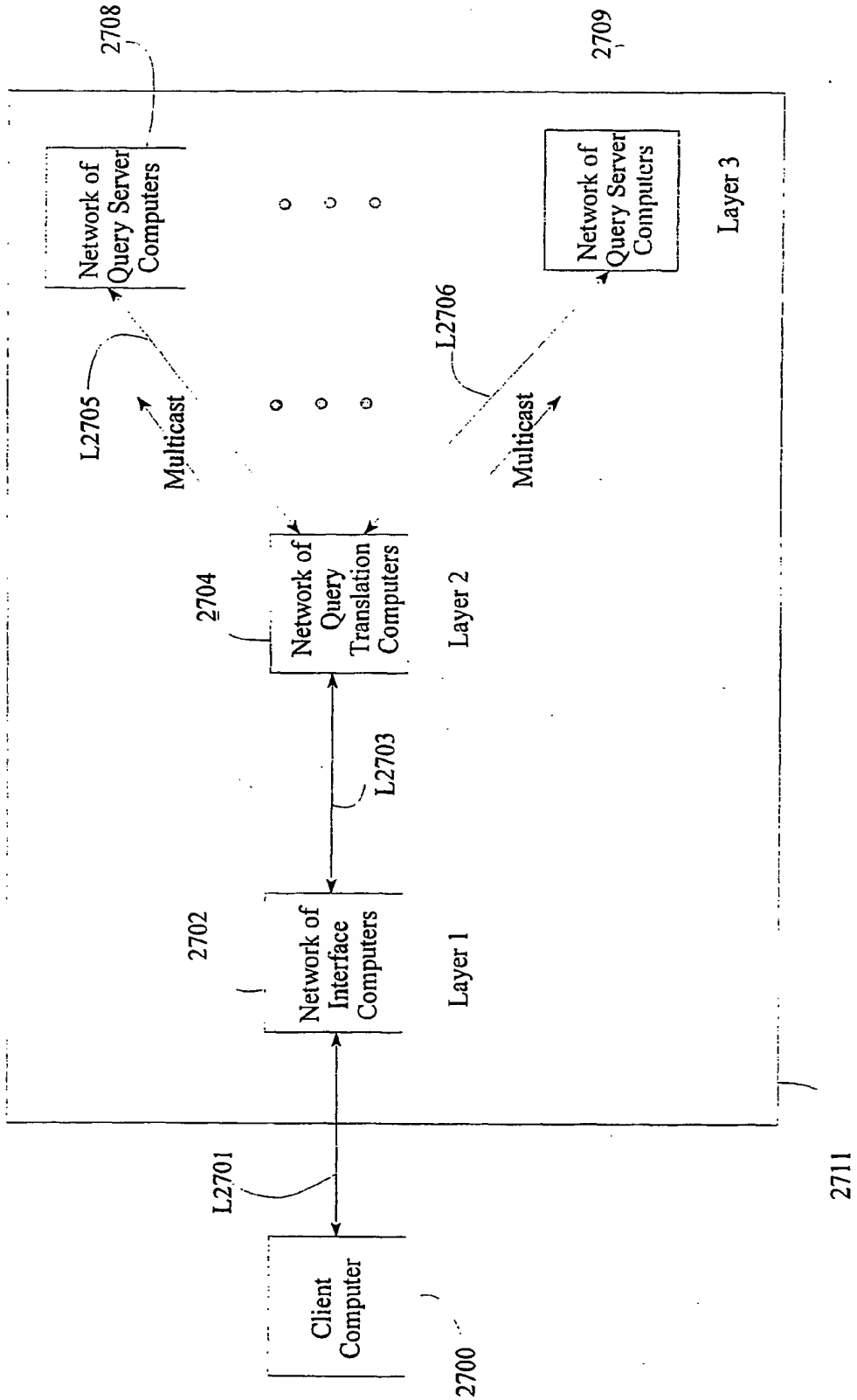
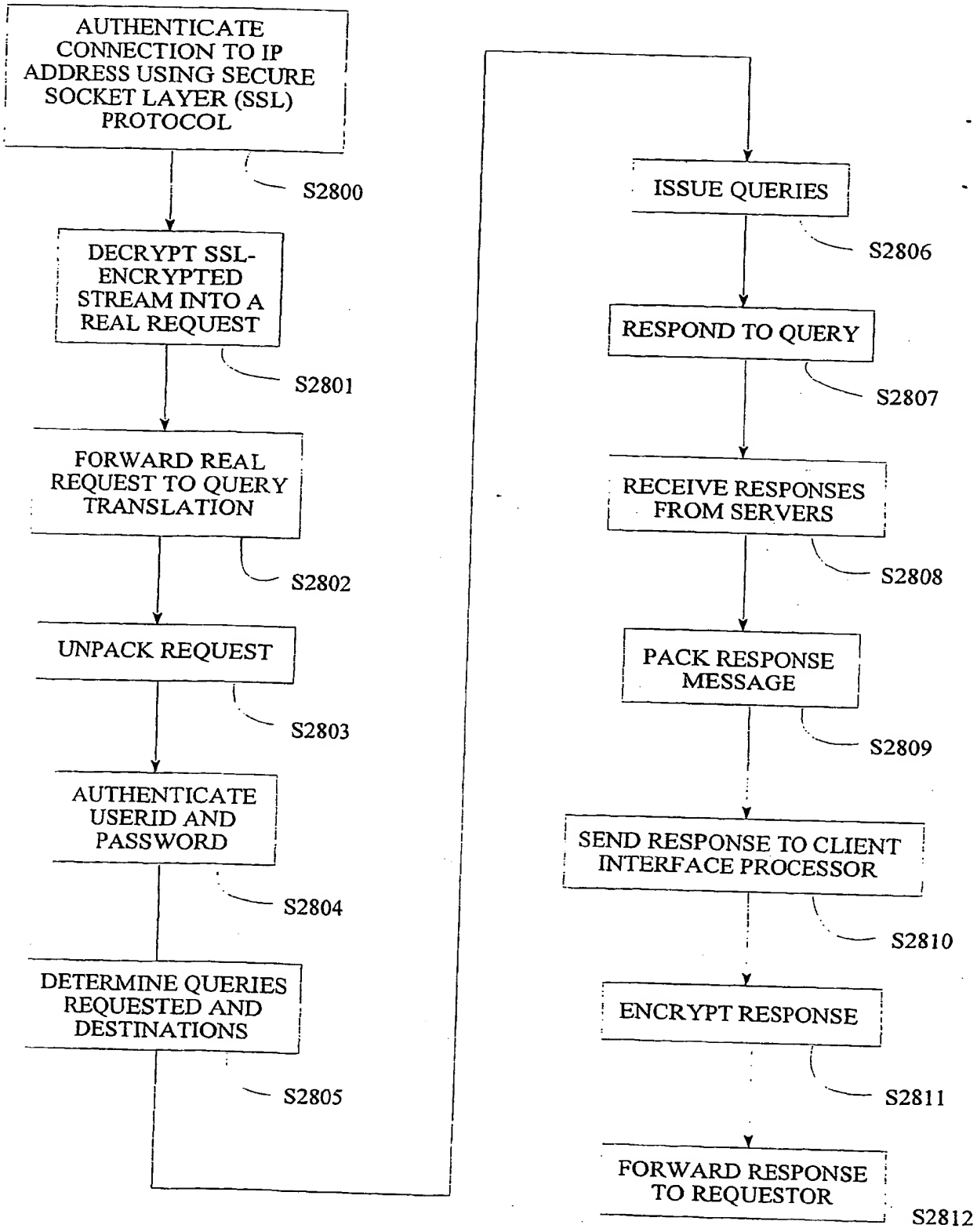

2708
2709
Network of Query Server Computers
Network of Query Server Computers
Layer 3
L2705
Multicast
L2706
Multicast
2704
Network of Query Translation Computers
Layer 2
L2703
2702
Network of Interface Computers
Layer 1
L2701
Client Computer
2700
2711

Fig. 47

Fig. 48


AUTHENTICATE CONNECTION TO IP ADDRESS USING SECURE SOCKET LAYER (SSL) PROTOCOL
S2800
DECRYPT SSL-ENCRYPTED STREAM INTO A REAL REQUEST
S2801
FORWARD REAL REQUEST TO QUERY TRANSLATION
S2802
UNPACK REQUEST
S2803
AUTHENTICATE USERID AND PASSWORD
S2804
DETERMINE QUERIES REQUESTED AND DESTINATIONS
S2805
ISSUE QUERIES
S2806
RESPOND TO QUERY
S2807
RECEIVE RESPONSES FROM SERVERS
S2808
PACK RESPONSE MESSAGE
S2809
SEND RESPONSE TO CLIENT INTERFACE PROCESSOR
S2810
ENCRYPT RESPONSE
S2811
FORWARD RESPONSE TO REQUESTOR
S2812

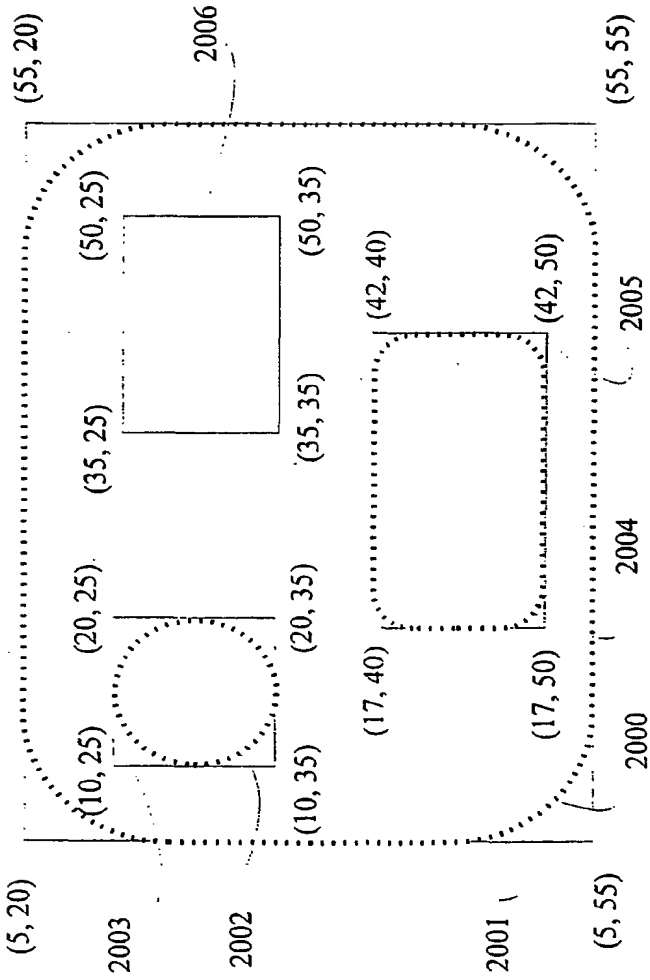

(55, 20)
2006
(55, 55)
(50, 25)
(50, 35)
(42, 40)
(42, 50)
2005
(35, 25)
(35, 35)
(20, 25)
(20, 35)
2004
(17, 40)
(17, 50)
(10, 25)
(10, 35)
2000
(5, 20)
2003
2002
2001
(5, 55)

Fig. 49

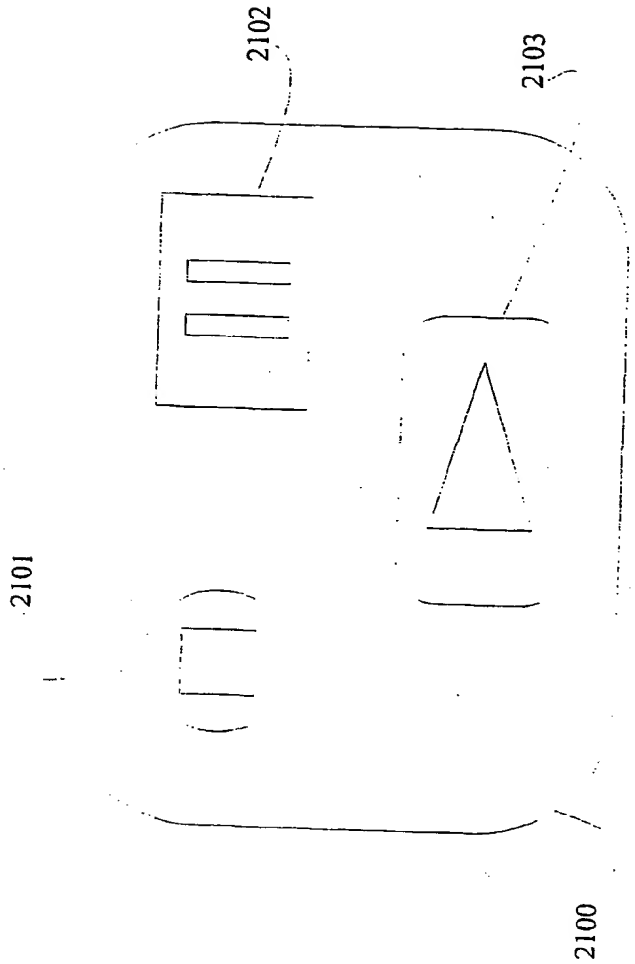


Fig. 50

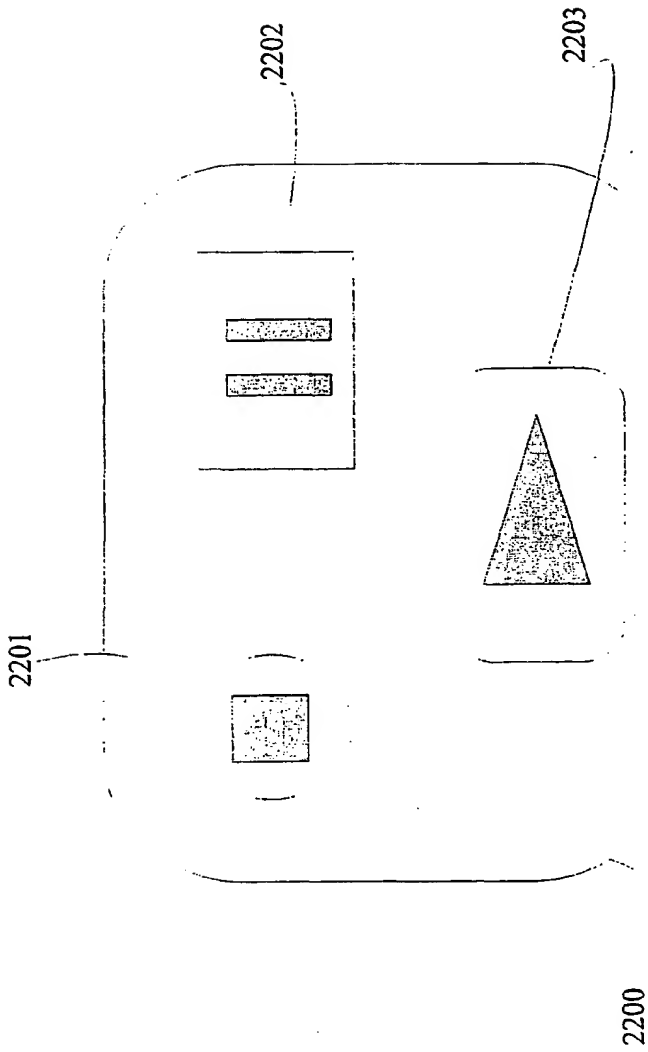


Fig. 51

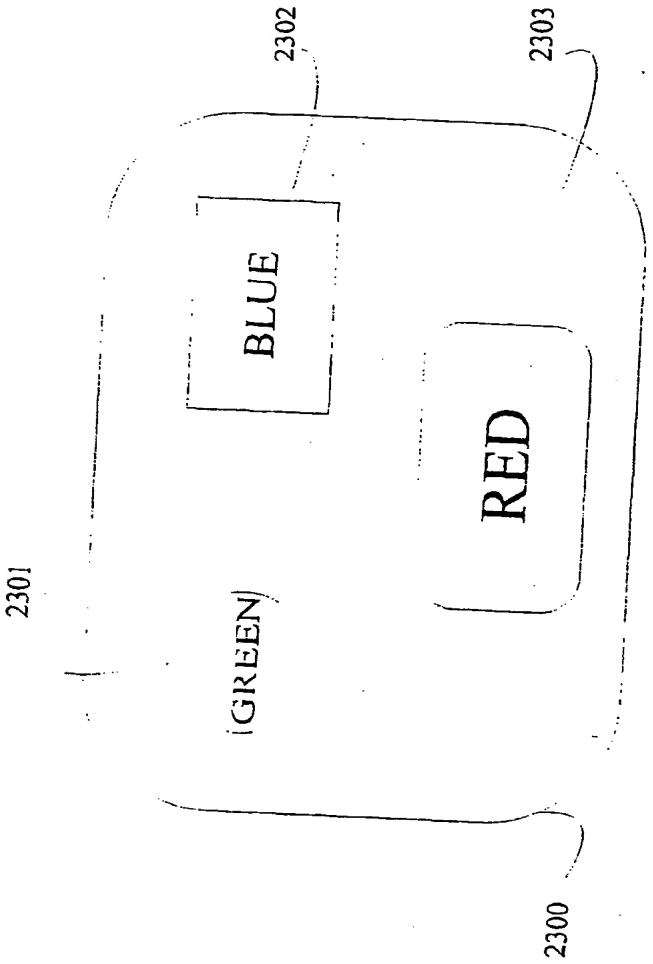


Fig. 52

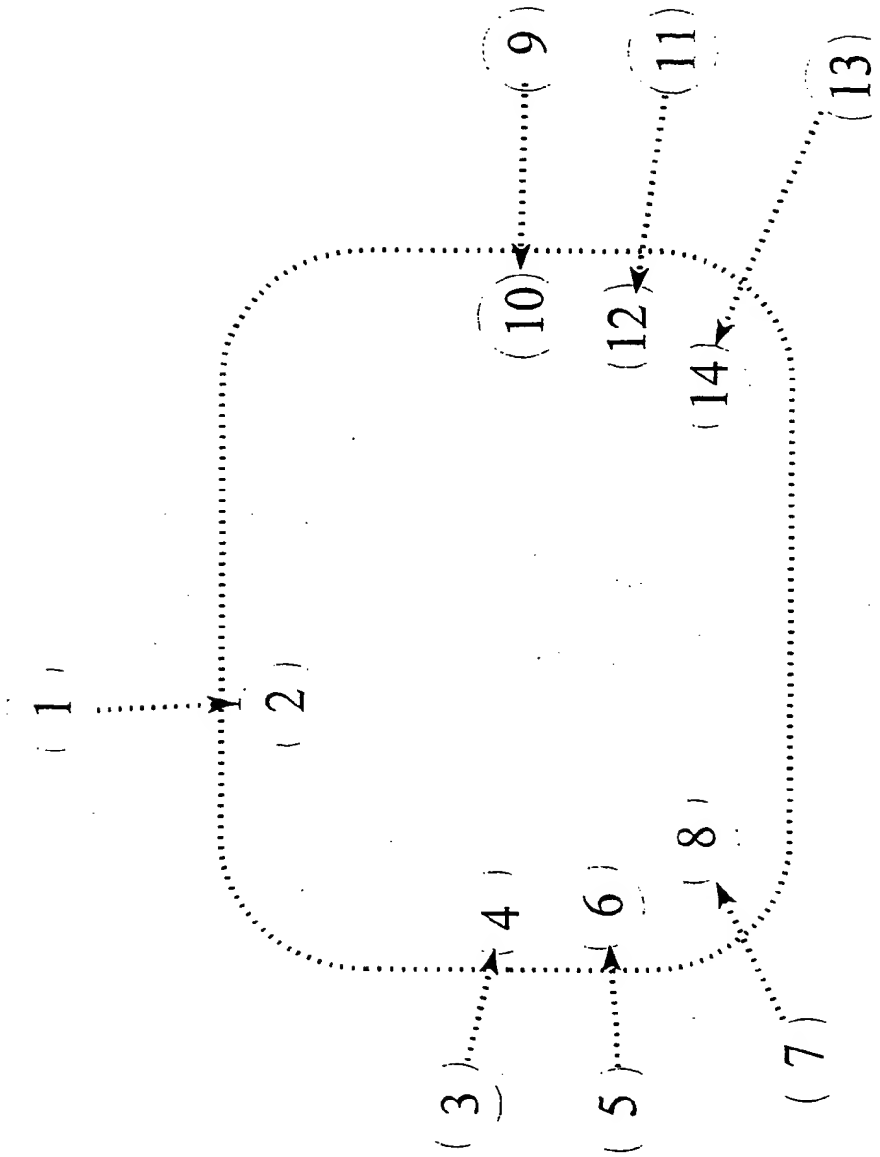


Fig. 53

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()_+-=,.<>/\;:"'

Fig. 54

Fig. 55

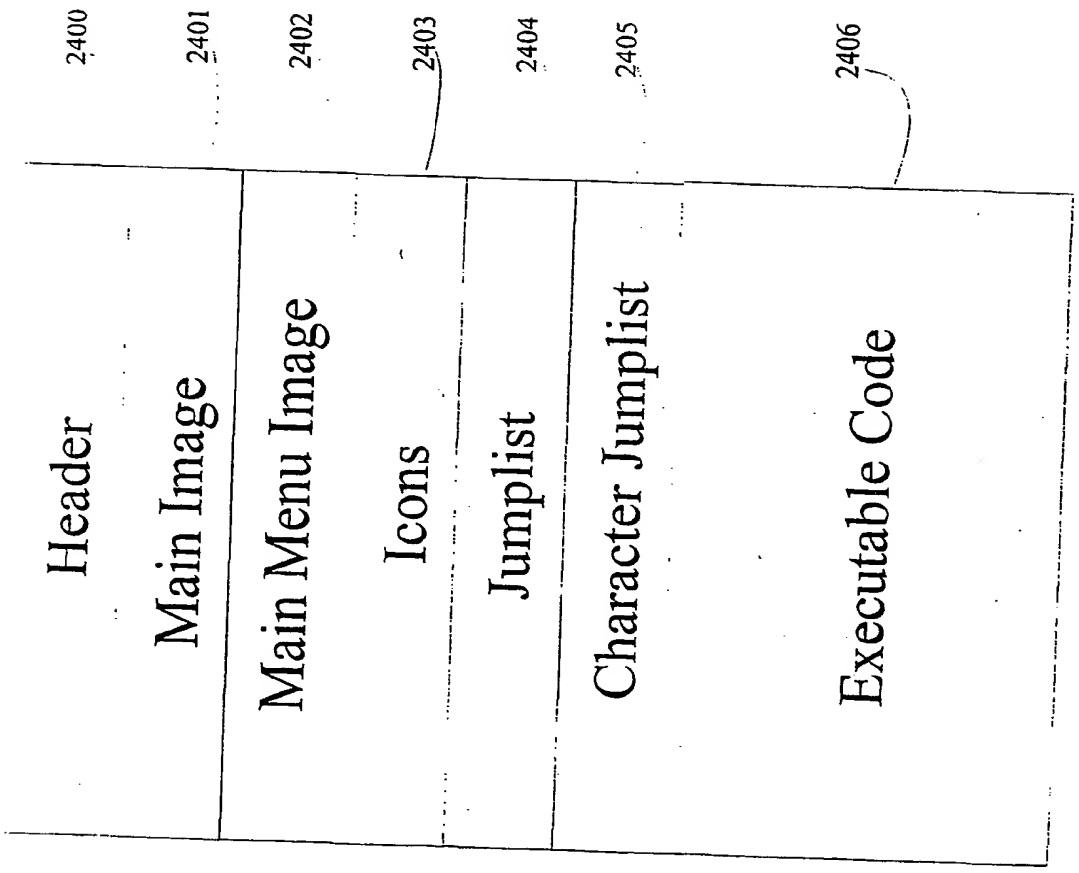

2400
2401
2402
2403
2404
2405
2406
Header
Main Image
Main Menu Image
Icons
Jumplist
Character Jumplist
Executable Code

Fig. 56

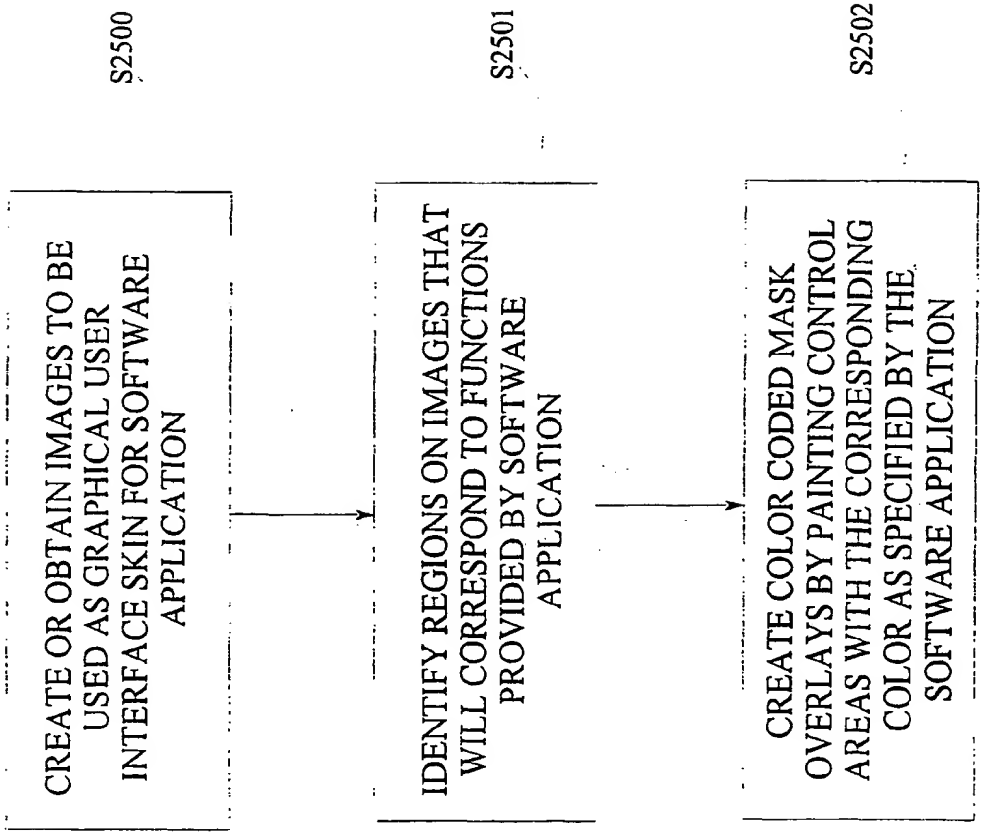

S2500
CREATE OR OBTAIN IMAGES TO BE USED AS GRAPHICAL USER INTERFACE SKIN FOR SOFTWARE APPLICATION
S2501
IDENTIFY REGIONS ON IMAGES THAT WILL CORRESPOND TO FUNCTIONS PROVIDED BY SOFTWARE APPLICATION
S2502
CREATE COLOR CODED MASK OVERLAYS BY PAINTING CONTROL AREAS WITH THE CORRESPONDING COLOR AS SPECIFIED BY THE SOFTWARE APPLICATION

Fig. 57

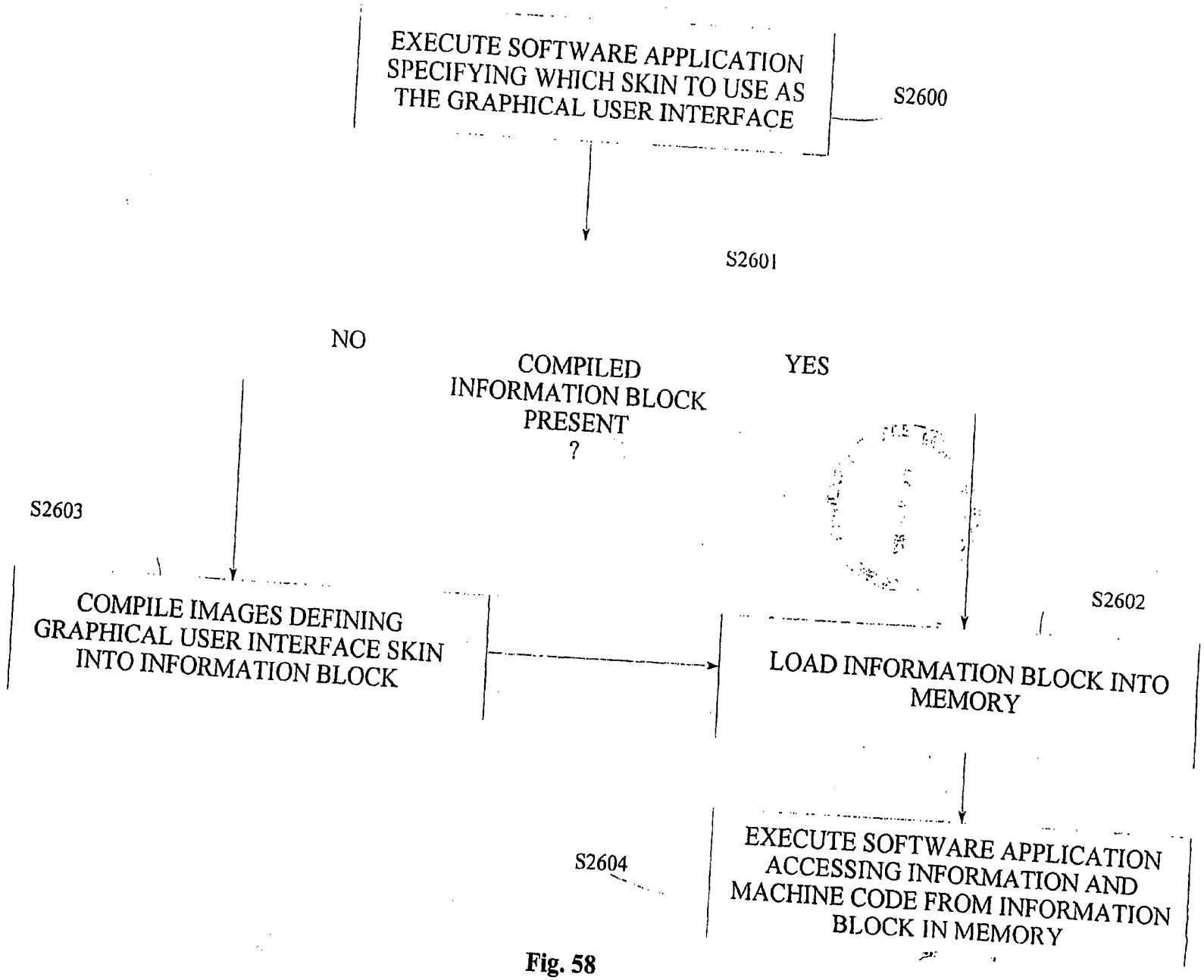

EXECUTE SOFTWARE APPLICATION SPECIFYING WHICH SKIN TO USE AS THE GRAPHICAL USER INTERFACE
S2600
S2601
COMPILED INFORMATION BLOCK PRESENT ?
NO
YES
S2603
COMPILE IMAGES DEFINING GRAPHICAL USER INTERFACE SKIN INTO INFORMATION BLOCK
S2602
LOAD INFORMATION BLOCK INTO MEMORY
S2604
EXECUTE SOFTWARE APPLICATION ACCESSING INFORMATION AND MACHINE CODE FROM INFORMATION BLOCK IN MEMORY

Fig. 58

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/00468

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :G06F 17/30
US CL :707/104, 1

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 707/104, 1, 204, 2, 9, 10; 709/223, 224

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

WEST, CAS ONLINE

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 6,012,068 A (BOEZEMAN ET AL.) 04 JANUARY 2000, See figs 1 and 2.	1-5, 11-14

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 13 JULY 2001	Date of mailing of the international search report 03 AUG 2001
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer SANJIV SHAH Telephone No. (703) 308-0956

Form PCT/ISA/210 (second sheet) (July 1998) ★

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/00468

Box I Observations where certain claims were found unsearchable (Continuation of item 1 of first sheet)

This international report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box II Observations where unity of invention is lacking (Continuation of item 2 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

Please See Extra Sheet.

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:
1-5, 11-14

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest.

☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US01/00468

BOX II. OBSERVATIONS WHERE UNITY OF INVENTION WAS LACKING
This ISA found multiple inventions as follows:

This application contains the following inventions or groups of inventions which are not so linked as to form a single inventive concept under PCT Rule 13.1. In order for all inventions to be searched, the appropriate additional search fees must be paid.

Group I, claim(s) 1-5, 11-14, drawn to method for managing media through a set of mechanism distributed between a client and a server system.
Group II, claim(s) 6-10, and 17, drawn to media managment system in audio player environment comprising analog signature.
Group III, claim(s) 15 and 16, drawn to media input/output device.
Group IV, claim 18, drawn to database management system.
Group V, claims 19-37, 38-43, 45, and 46, drawn to method for delivering commercials to a computer.
Group VI, claims 47 and 48, drawn to a memory for storing information for delivering commercials to a computer comprising a data structure.
Group VII, claim 49, drawn to a system for providing redundant storage for information.
Group VIII, claim 50, drawn to a method of querying a plurality of servers with a single query.
Group IX, claim 51, drawn to a method for skinning a graphical user interface of a software application.

The inventions listed as Groups I through X do not relate to a single inventive concept under PCT Rule 13.1 because, under PCT Rule 13.2, they lack the same or corresponding special technical features for the following reasons:
Group I has a a special technical feature of set of mechanism distributed between server and client which is lacking from Groups II to IX.
Group II has a special technical feature of media data player with analog signature which is lacking from Groups I, III to IX.
Group III has a special technical feature of a media input/output device which is lacking from Groups I-II and IV to IX.
Group IV has a special technical feature of a database management system which is lacking from Groups I-III and V-IX.
Group V has a special technical feature of delivering commercials to a computer which is lacking from Groups I-IV and VI-IX.
Group VI has a special technical feature of a memory storing commercial information which is lacking from Groups I-V and VII-IX.
Group VII has a special technical feature of a redundant storage information which is lacking from Groups I-VI and VIII-IX.
Group VIII has a special technical feature of querying the plurality of servers which is lacking from Groups I-VII and IX.
Group IX has a special technical feature of a graphical user interface which is lacking from Groups I-VIII.

THIS PAGE BLANK (USPTO)